
SEGA®
COMPUTER
The Official Sega User Club Magazine
APRIL ISSUE 1985
ARTICLES
Complete Ram Memory Map
Program Disection
PROGRAMS
Space Ace
Grid Runner
Graphics Screen Printer/ Plotter Dump Truck Routine
Flip!
Big Characters
Roman Numerals
Leo Lion
Strings

MORE THAN FIFTY PROGRAMS

By MICHAEL HOWARD

So you want some more programs to key into your Sega? Well, this book has over Fifty for you to tap out covering everything from games to education, machine code and basic, short and long. (all programs will run on a 16k or 32k cartridge). Pages of programs, each one disected and explanations of how and why it works!

INTRODUCTION

Dear Reader

This issue of your favourite micro magazine is tilted ever so slightly in favour of the "serious computer buffs" amongst our membership, which I might add is growing at a staggering rate of fifty new members per month. A very grafifying fact for those of us who have the task of producing all the mindless drivel that goes in between the front and back covers.

But seriously folks in this issue you will find articles on machine code by Michael Howard and Colin Smith, utility programmes to type in plus a long awaited review on some excellent disk based business software. Also a lighthearted account of our recent trade launch on board the Amstrad Express.

Keep your eyes on the goggle box these coming weeks and watch out for our Rowan Atkinson adverts the guy is pure majic and his rubber faced expressions are sure to help our cause.

May I take this opportunity to welcome all new members to our illustrious ranks and wish you all happy reading.

Steven Kenyon

Contents

LOCAL SEGA USERS CONTACTS

PUKEKOHE SEGA USERS CLUB
C/o 4 Roose Avenue
Pukekohe
Contact: Selwyn
Ph. Pukekohe 86-583

AUCKLAND CENTRAL SEGA USERS CLUB
C/o 287 Broadway Furniture
Newmarket
Contact: George Shaw
Ph. 547-543

ROTORUA SEGA USERS CLUB
C/- 61 Devon Street
ROTORUA

TOKOROA SEGA USERS GROUP
C/o 1 Pio Pio Place
TOKOROA
Contact Geoff Phone Number 67-105
Tokoroa

SOUTH TARANAKI MICROCOMPUTER SOCIETY
D.M. Beale
7A Clive Street
HAWERA

NAPIER SEGA USERS CLUB
Sec E.P. Lins
41 Higgins Street
NAPIER

GISBORNE AREA USER'S CLUB
Trevor Gardiner
Ph. 63-068 HM
or 87-175 WK

HAMILTON SEGA USER'S CLUB
P.O. Box 1548 Hamilton
Contact Leslie Wong
Ph. 384-892 Ext 66
Meetings fornightly in room KG09
University of Waikato

CHRISTCHURCH USER'S CLUB
Contact Graham Rudman
29 Primrose Street
CHRISTCHURCH 5

The above are contact names and addresses for Sega Users Clubs. If you wish to have your club advertised write to Sega Users Club P.O. Box 2353, Auckland.

If you have set up a local area user's club and you would like us to publish the details concerning your club please send them to us and we will publish the information for no charge.

READER'S LETTERS

DEAR EDITOR

Please include the following in your magazine as a brief account of the South Taranaki Sega Clubs activities for the past ten months. In June of last year an inaugural meeting was called to form a Micro Computer Club for South Taranaki, by J. Callaghan, Computer Tutor at our local High School. This was in response to those people who had attended Night School earlier in the year. At this meeting it was decided individual needs would be best served by dividing into individual groups, thus the South Taranaki Sega User Group was formed. This format serves us well, as every two months all Computer owners get together (and argue who has got the better deal). From this a greater knowledge of how different Computers work is gained and topics such as program conversion discussed.

The Sega section had it's first meeting in August of last year. Have you ever tried to start a club? It is HARD! My first advice to anyone starting a Club, is to listen. I had preconceived ideas of what the Club should do, this is grossly wrong. The basis of any Club should be to serve all its members if possible. The first meeting went off badly, the second I sat and listened. Actually I found the best method was not to be formal, but to put a program on the screen and hash it around. Ideas come from all directions. In future we intend to run a new game or program each meeting for members evaluation, this could save a lot of time trying to run it in a shop. Software demonstrations must really bug a shop assistant, especially if the shop is busy.

Secondly we have found a very real need for further education both in basic and machine code. In the early part of 1985 we intend to run a basic course covering all aspects of the Sega, including Graphics, Sound, Basic Command and program designing. If there is anyone out there who would be interested in helping out with a Machine Code class, please, please give me a call (even I need help in this area). If you wish to enrol for either of these classes, phone me now. Classes will be held weekly at Farmers Mutual Conference Room, Regent Street, Hawera. Our monthly meetings are held on the 4th Wednesday of the month at the above address. Cost is about $5.00 per year to cover Hall hire. Remember a Club is only as good as its support and we need more members. I feel there are a lot of people out there who do not join a Club for fear of being shown up or being put to shame by the so called "WHIZZ KIDS". All I can say is Hogg Wash, a lot of our members are parents who brought their Computer for their kids and want to learn so they can have fun teaching the kids. One year ago a Ram was a sheep and a Variable was a sea breeze to me'. . .

To close I would like to tell of something that happened just prior to Christmas, the young daughter of a friend was around, since she will be taking typing next year she was keen to try out the Typing Tutor Programme, after setting the program to run, away she went. The first word came up, a shorty of about five letters BLEEP BLEEP BLEEP "Typing speed 5 words per minute", the next word came about the same, then came a real whopper DEXIO- something or other, she couldn't say it let alone type it. Unperturbed away she went BLEEP BLEEP BLEEP, it must have taken all of 45 seconds, then to my amazement the computer came back with a typing speed of 40 words per minute.

"Hey look mum, 40 words per minute". My only explanation is a program error and the Computer should have come back with a spead of ".40" words per minute. They now own a Sega and mum has high expectations for her "Whizz Kid" daughter.

Thats all for now,
Dave Beale,
Sega User Group,
Hawera.
Phone HWR 85108 evenings.

EDITORS REPLY

Thank you for your support. We are looking for people in every region around New Zealand to help set up local area Users Club's. We will publish meeting dates and times as well as contact names and addresses in our magazine. If anybody is interested in forming or has formed a Users Group please contact us.

DEAR EDITOR

As a very recent purchaser of the SEGA 3000 I have found the Users Club and the Users Club Magazine of great value. Keep up the good work.

Many thanks for the Sprite Editor and the Music Editor cassettes.

Another query. In Championship Golf — does any one else get the "Days record score" being given as the losers score e.g. +20 etc.? In golf the record should be the lowest score. Apart from this it's a great game.

In the last Magazine you published Glenn Howard's "Gunslinger." Lovely graphics! The game itself became rather quickly boring because all one needed to do was count 15 steps then fire. My son and I have it being more challenging by adding line 2025 RM=Int(Rnd(8)*58)+162 and changing line 2270IFx<RM THEN 3280. This gives a random DRAW time of from 1 to 15 steps.

Keep up the good work and keep all we SEGA users up-to-date and keenly trying out new things.

Sincerely yours
Alan Morgan

EDITORS REPLY

Thank you for your kind words about the magazine (at least somebody likes us!)

We have noticed what you have mention about the Champion Golf cartridge. This cartridge was written in Japan (this is a prime example of Japanese logic). Unfortunately nothing can be done about this.

DEAR EDITOR

The FIREMAN game (November/December '84) was well received in this household, but we thought a couple of improvements were well worthwhile making:

1. The main loop (lines 11Ø to 39Ø) can repeat a maximum of 2ØØ times and then the game will end (i.e. when FC=2ØØ in line 37Ø), during the course of one stage. You can get more adrenalin into your bloodstream if you can see your time running out! Add the following lines:

```
382 IF FC<1 THEN GOTO 386
384 BLINE (2ØØ,16Ø)-(231,167),15,BF
386 CURSOR 2ØØ,16Ø:COLOR
    1:PRINT 2ØØ-FC
895 CURSOR 2ØØ,152:COLOR
    6:PRINT"COUNTDOWN"
```

2. If you would like to see just how

cont. on p22

THE AMSTRAD EXPRESS
a lot on the line

Following on from our results articles of the Jan/Feb issue, our launch campaign for Amstrad got underway on March 18th, on platform one of Auckland central station.

The theory was if you can't be sure the dealers can come to your showroom. Then take your showroom to the dealers (apparently the idea first emerged with some chap called Mohamet). First step was to involve the railways to lease a carriage and paint the exterior silver so as everyone would see it coming.

Next fit the whole thing out with carpet, display stands for Sega and Amstrad, projector and screen for our presentation, tables and chairs for our dealers, spotlights so we don't stumble in the glow of TVs, and most important of all a bar and fridge for light refreshment.

The launch in general was a terrific success. Many dealers who last year were unconvinced about Sega were surprised and impressed with the tremendous advances made over the 12 months since Sega was launched, and consequently are now supporting it, Amstrad was also well received and is sure to take a large share of the $1,000 and market this year.

The whole campaign went off with alarmingly few incidents.

Our first port of call was Rotorua, where our team T shirts (a rather bold blue and white stripe), were aired publicly for the first time, and caused quite a stir amongst the staff and clientell of the local Cobb & Co. Imagine our delight when a bottle of wine arrived courtesy of the management for entertaining the guests, and imagine our horror when we were then invited to ride the mechanical bull in the house bar which just happened to be operating there that night. Yes we all had a try, and all have the bruises on our inner thighs as witness to the occasion.

Palmerston North as our second port of call was another great success, and gave myself and Aldas (a new member of the team) a chance to practice breaking and entering, as the door to our motel refused to respond to the key and the window remained the only form of entry.

Our journey to Wellington provided our first bid to become driver of the year, as one who shall be nameless, altered

cont. on p12

Program Dissection

Program by Nigel Irwin, Wainuiomata **Dissection by Philip Bachier**

Here is a clever little program that makes use of the SEGA's sound, sprite and graphic capability. Not bad at all when you consider it only uses just over 1K. As you can see its very easy to construct your own simple games. This program doesn't score but you can add this yourself. . .

EDITORS NOTE: This waffling went on for about another two pages and ended up on a completely different and irrelevant subject. If given the chance this article probably would have filled the entire magazine.

Line 1Ø:
CLS clears the screen the computer is currently working on. (In this case the text screen.) The colour is set to red writing (6) upon a white background. (15) The CURSOR statement moves the cursor to 14 characters along the screen and 6 down where the message is printed.

Line 2Ø:
This underlines the heading, prints two blank lines and part of the instructions.

Line 3Ø:
The rest of the instructions.

Line 4Ø:
This counts the variable L from 1 to 6ØØ. This causes a delay while the computer is counting.

Line 5Ø:
The SCREEN 2,1 statement keeps the text screen on the monitor, however it passes all information to be printed etc to the GRAPHICS screen. In other words it allows you to update the graphics screen without having to look at the alterations being made. You remain on the text screen. The CLS statement (see Line 1Ø above) does not clear the text screen but the graphics screen.

Line 6Ø:
This colours the entire graphics screen including the top and bottom areas in black (colour code 1. See page 1ØØ of the Level III manual).

Line 7Ø:
This is a start of a FOR-NEXT loop. The computer will now count variable S from 1 to 1ØØ. S is increased by one every time the computer sees the statement NEXT S (see Line 1ØØ).

LiNE 8Ø:
The RND function creates a random number between Ø and 1. This is then multiplyed by 255 to give a random number between Ø and 255. This random number is variable A. (see Line 1ØØ).

Line 9Ø:
D is made a random number between Ø and 191.

Line 1ØØ:
The PSET command places a dot on the screen at position A along and D down. This positioning is random (see LineS 8Ø and 9Ø above) so little white dots ("stars") will appear in places all over the screen. The NEXT S statement is the end of the FOR-NEXT loop. If S is not 1ØØ then the computer jumps back up to line 8Ø. This means that we will get 1ØØ stars on the graphics screen.

Line 11Ø:
Switches back to the text screen

Line 12Ø:
This line makes a sound effect (stunning isn't it!?) and puts the message on the text screen at location 8,2Ø.

Line 13Ø:
The INKEY$ function scans the keyboard. What this line does is that if nothing has been pressed it will go back and scan the keyboard until a key is pressed.

Line 14Ø:
A key has been pressed so the computer jumps to this line and switches to the graphics screen.

Line 15Ø:
Z is the x position (along) for your craft. C is the position down the screen for the alien. M is the speed of the alien.

Line 16Ø:
This defines the top left hand block of your ship. NB there are four blocks making up this sprites as it is a MAG 1 sprite (see Line 24Ø).

Line 17Ø-19Ø:
The other three blocks making up your ship.

Line 2ØØ:
The PATTERN statement defines the explosion.

Line 21Ø:
The Alien craft.

Line 22Ø:
This sets the alien craft at the top of the screen.

Line 23Ø:
X is a random number between 1Ø and 25Ø. This is the X coordinate (along the screen) of the alien craft.

Line 24Ø:
MAGI tells the computer that all the sprites are made up of four 8x8 pixel sprites (i.e., a 16x16 pixel sprite. The SPRITE statement places a sprite on the screen. In this case it is sprite number zero (the most prominent sprite) at position Z along and 16Ø pixels down the screen. The shape of the sprite is gotten from patterns Ø-3 and the sprite is green (colour code 12. See page 1ØØ of the Level III manual).

Line 25Ø:
Sprite number 4 is positioned at X,C and is the alien craft.

Line 26Ø:
Now we get to the nitty-gritty. The computer scans the keyboard and places the key pressed in Z$. C (the Y position of the alien) increased by M moving the alien craft down the screen.

Line 27Ø:
This tests to see if the left arrow key has been pressed. See page 18 and 19 of the Level III manual. If this arrow key has been pressed Z is decreased by 3. Z is the X coordinate of your ship.

Line 28Ø:
If the right arrow key has been pressed your ship is moved 3 pixels to the right.

Line 29Ø:
This checks to see if the space bar has been pressed. If it has a red line is drawn up the screen. The computer is told to GOSUB 34Ø. The computer jumps to line 34Ø and will work through the program there until it comes across the word RETURN. When the word RETURN is found the computer jumps back to the end of line 29Ø.

We will say that the space key has been pressed so that the computer jumps to line 34Ø.

Line 340:
This is the Fire routine. This erases the line drawn on Line 290.

Line 350:
Sound channel 2 is turned on with a frequency of 110 hertz (this is a very low pitched sound), with a volume of 15 (the loudest volume).

Line 360:
SOUND0 turns all sound off. This gives very short quick click.

Line 370:
The computer sees the word RETURN so it jumps back to Line 290.

Line 300:
If the alien is more than 60 pixels down the screen it is told to speed up and move 2 pixels instead of 1 (M=2).

Line 305:
If the alien is more than 100 positions down the screen M is set to 3. M is the speed of the alien. (The number of pixels it moves down the screen each time it moves)

Line 307:
This tests to see if the alien craft has reached your level. If it has the game ends.

Line 310:
This is ever so slightly complicated so I shan't try to explain it as I will end up waffling and confusing myself. This line tests to see if the space key has been pressed (you have fired at the alien craft) and that you have hit the alien. If you have hit the alien the computer jumps to line 380.

Line 380:
The hit routine. You have just managed to destroy an alien craft. The computer places a red explosion (sprite 5 with a colour of medium red) underneath the alien craft at position X,C.

Line 390:
C, the y coordinate of the alien is set to 1. This is the top of the screen.

Line 400:
The next 5 lines make an explosion sound effect. F is the volume of the bang. This is stepped from 15 (the loudest) to 1 (the softest).

Line 410:
This is the line that makes the noise. Channel 4 which is white noise is turned on at volume F.

Line 420-430:
D is a delay. It slows the sound effect down so that you can hear it. This is also the end of the FOR-NEXT loop decreasing the volume of the sound effect.

Line 440:
Turns all the sound off.

Line 450:
This erases the explosion by changing its colour to transparent i.e., we see straight through it. But it is still there.

Line 460:
Tells the computer to jump back to line 230 and start and set up a new alien craft.

If you had not hit the alien craft the computer would have carried onto Line 320.

Line 320:
This checks to see that you have not fallen off the left hand side of the screen. If Z is less than 1 i.e., your about to disappear then Z=0 which is the side of the screen.

Line 325:
Tests to see you are not going of the right hand side of the screen. The figure of 240 if used as your ship is 16 pixels long. If 255 was used you would not be able to see your ship.

Line 330:
The programme goes back to line 240 and moves the sprites.

```
10 CLS:COLOR6,15:CURSOR14,6:PRINT"SPACE ACE"
20 CURSOR13,7:PRINT"~~~~~~~~~~~~~":PRINT:PRINT"  USE LEFT AND RIGHT CURSOR KEYS TO
    MANOVER YOUR SPACE SHIP BENEATH THE"
30 PRINT"ALIEN AND PRESS THE SPACE BAR TO FIRE"
40 FORL=1TO600:NEXTL
50 SCREEN2,1:CLS
60 COLOR1,1,(0,0)-(255,191),1
70 FORS=1TO100
80 A=INT(RND(1)*255)
90 D=INT(RND(1)*191)
100 PSET(A,D),15:NEXTS
110 SCREEN1,1
120 BEEP:CURSOR8,20:PRINT"PRESS ANY KEY TO BEGIN"
130 IFINKEY$=""THEN130
140 SCREEN2,2
150 Z=127:C=1:M=1
160 PATTERNS#0,"00010B0F0B060405"
170 PATTERNS#1,"05050FBFFFBF7FE3"
180 PATTERNS#2,"0000A0E0A0C04040"
190 PATTERNS#3,"4040E0FAFEFAFC8E"
200 PATTERNS#5,"158795345ADEBC98"
210 PATTERNS#9,"815A3C66663C5A81"
220 C=1
230 X=INT(RND(1)*240)+10
240 MAG1:SPRITE0,(Z,160),0,12
250 SPRITE4,(X,C),9,4
260 Z$=INKEY$:C=C+M
270 IFZ$=CHR$(29)THENZ=Z-3
280 IFZ$=CHR$(28)THENZ=Z+3
290 IFZ$=" "THENLINE(Z+7,161)-(Z+7,1),8:GOSUB340
300 IFC>60THENM=2
305 IFC>100THENM=3
307 IFC>160THENSCREEN1,1:CLS:PRINT"BANG!!! YOUR DEAD.":END
310 IFZ+7>XANDZ+7<X+8ANDZ$=" "THENGOSUB380
320 IFZ<1THENZ=0
325 IFZ>240THENZ=240
330 GOTO240
340 BLINE(Z+7,161)-(Z+7,0)
350 SOUND2,110,15
360 SOUND0
370 RETURN
380 SPRITE5,(X,C),5,8
390 C=1
400 FORF=15TO1STEP-1
410 SOUND4,2,F
420 FORD=1TO20
430 NEXTD,F
440 SOUND0
450 SPRITE5,(X,Z),5,0
460 GOTO230
```

How To Program In Machine Code Language On The Sega SC3000

by COLIN SMITH

You will need to refer back to table 2 in the last issue.

Implied Addressing.

Interrupt Register (I).

When the computer is first turned on, the contents of this register are automatically set to zero. Therefore it is necessary to load the high byte address into the register, as the program starts to run. This can only be achieved by loading the Accumulator (A) with the high byte address and loading the opcode EDH,47H. If it is required to check the contents of the register it is possible to load the date into the A register using the code EDH,57H.

	Addr	Hex	
Program Counter	AØØØH AØØ1H	EDH 47H	Two Byte Opcode.

	REG I	REG A
Before	ØØH	5CH
After	5CH	5CH

Refresh Register (R),

This address contains the low byte address of the memory being refreshed. As with the interrupt register it can only be read and written too by the A register. This register is of little use to the programmer and data can be lost from memory by writing the same address too often and not allowing the rest to be refreshed.

Register Indirect Addressing.

(When used as source)

Their codes specify a register pair to be used as a pointer to any location in memory. If we require to obtain data from location CØCØH and load it into the A register, this is what we would do. One of the register pairs (HL,DE,BC) would be loaded with the address CØCØH and then the appropriate opcode would be given to the computer. The contents of the specified memory location would then be copied into the A register.

	Addr	Hex	
Program Counter	AØØØH	1AH	Opcode.

	Mem CØCØH	Reg A
Before	52H	7FH
After	52H	52H

All three register pairs can load data into the A register, but only register pair HL can load into all the registers, including to itself. These last two codes (66H,6EH) may seem a little confusing, but the answer lies in the way the computer times its operations. When the computer reads the opcode it knows it has to fetch data from memory and load it into a register. With the last two codes it knows the address is stored in registers HL and reads the address from them. Having read the address the contents of the HL registers are no longer needed, so there is no reason why the data cannot be copied into them.

(When used as destination)

It is basically the reverse of the above. If we wanted to load the contents of a register into CØCØH, we would load the appropriate register pair with the address and then load the opcode to the computer. The contents of that register would then be copied into the memory address.

	Addr	Hex	
Program Counter	AØØØH	Ø2H	

	Memory	Reg A
Before	CØCØH 52H	7FH
After	CØCØH 7FH	7FH

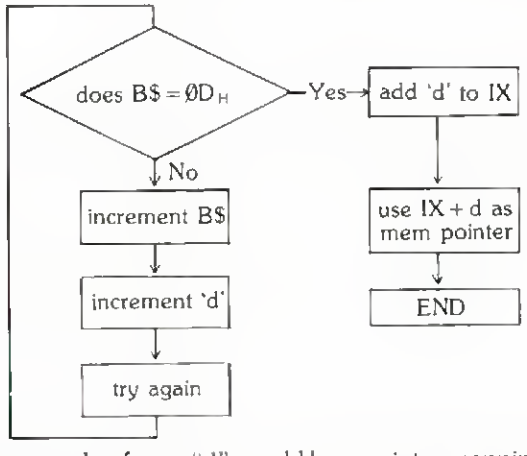
However, if we look at the two codes that load the H and L registers, we end up loading part of the address into the address itself. Unlike the above case, once the address was read the data in the registers is still required. These two codes are of little use to the programmer.

Indexed Addressing.

The two index registers IX and IY are normally loaded with data from table 3. These registers are then used as bases on which to access the memory by defining a displacement value 'd' which is added to the base value. This new value is then used to point to an area of memory. These are very useful when operating on tables of data and is the reason why two registers are provided as it is often required to operate on more than one table of data.

They would be operated in a similar manner to this:-

1. A base value will have been loaded into the IX register lets say BØØØH.
2. There would be some form of data processing take place. flow chart



B$ is a sample of text, "d" would be a pointer scanning along B$. All text ends in ØDn.

Both B$ and 'd' would be incremented till B$ = ØD at this stage 'd' would be at a certain value, this would be added to IX by loading the opcode and placing 'd' as the operand.

3. If we assume that 'd' = 1FH, this would be added to BØØØH to make BØ1FH and this value would be used to point to the address from which one of the registers will be loaded.

	Addr	Hex	
Program Counter	A000H	DDH	Opcode
	A001H	7EH	Operand
	A002H	1FH	(displacement value)
Base value	B000H	05H	
+	::::	::	
displacement	B01FH	7CH	

	A reg
Before	54H
After	7CH

(When used as destination)

The reverse of the above is true, with 'd' being the displacement of the address to be written too.

To use IX or IY, you imagine your using HL, but just place DDH or FDH in front of operand respectively. Followed by displacement.

Extended Addressing.

(When used as source)

In the eight bit load group only the Accumulator ('A'-register) can be loaded or unloaded by extended addressing. It is a very direct way in which to move data. By using the code 4AH as the opcode, it is only necessary to write the address from which the data is to be read.

e.g., We wish to load the A register from 2C3D.

	Addr	Hex		
Program Counter	A000H	3AH	Opcode	Operand
	A001H	3DH	low byte	
	A002H	2CH	high byte	

	Addr	A Reg
Before	2C3D = 94H	F0H
After	2C3D = 94H	94H

(When used as destination)

When used as the destination the above is reversed and the contents of the A register are copied into the memory address.

Immediate Addressing.

This two byte opcode will load the register concerned immediately with the data following the opcode.

e.g., We require to load the E register with the data 1CH.

	Addr	Hex	
Program Counter	A000H	1EH	Opcode
	A001H	1CH	Operand

	E Reg
Before	02H
After	1CH

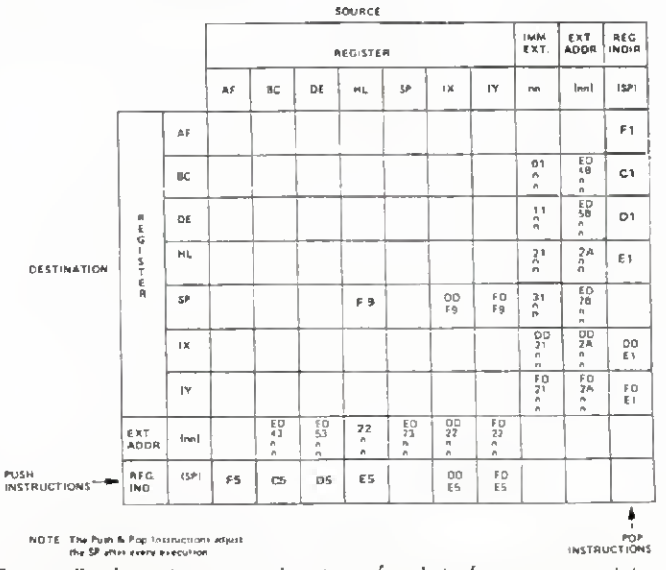
This mode of addressing can be used to load data immediately to a memory location. By using the HL register pair to point to the memory location, loading the code 36H followed by the data as the operand, this is very easily achieved.

It is also possible to load immediate data by using the index addressing mode.

e.g., The two index registers will have had their base values loaded into them, so by loading the two byte opcode, then the displacement 'd', followed by the data, this will be achieved.

	Addr	Hex		
Program Counter	A000H	FDH	Opcode	Operand
	A001H	36H		
	A002H	F9H	Displacement	
	A003H	01H	Data	

16 BIT LOAD GROUP TABLE 3



Generally there is no need to transfer data from one register pair to another. (It is possible and will be shown later). Looking at table 3 you will see that there are no codes to achieve this anyway. The three exceptions are used to load the Stack Pointer with its 16 bit address for the top of stack. One of the three registers (HL,IX,IY) is loaded with the address of where the stack is to be placed. The appropriate opcode is loaded and the contents of the registers are copied into the Stack Pointer register.

Immediate Extended Addressing.

These codes load the register pairs with the data following in the next two memory locations. The low byte first, followed by the high byte.

e.g., We wish to load the register pair DE with the code A000H.

	Addr	Hex		
Program Counter	C0C0H	11H	Opcode	Operand
	C0C1H	00H	To reg E	
	C0C2H	A0H	To reg D	

Before	D	32H	E	CFH
After	D	A0H	E	00H

Extended Addressing. (When used as source)

These codes load the register pairs with data held in memory. They are all two byte opcodes except for the HL register and the two bytes following contain the sixteen bit address where the data is stored. The low byte of the address first followed by the high byte.

The register pairs are sixteen bits wide where as the memory location contents are only eight bits wide. So to obtain sixteen bits of data, the eight bits of data found at the first memory location are used as the low byte of data. The address is then incremented, and the data found there becomes the high byte. e.g., Load HL from D0CCH.

	Addr	Hex		
Program Counter	A00H	2AH	Opcode	Operand
	A001H	CCH	Low Byte	
	A002H	D0H	High Byte	
	" "	" "		
	" "	" "		
	D0CCH	32H	Low Byte	Data
	D0CDH	FFH	High Byte	

Before	H	C0H	L	F0H
After	H	FFH	L	32H

(When used as destination)

cont. p12

Yet Another Game Which I Won't Try to Explain

GRID RUNNER

```
10 REM //       GRIDRUNNER       //
20 REM // BY N. CAMPBELL  1985 //
30 REM // FOR ANY SEGA SC-3000 //
40 GOTO430
50 IFJ=0THENGOSUB320
60 IFJ=1THENGOSUB220
70 FORP=1TO2
00 IFD(P)=0THENY(P)=Y(P)-1
90 IFD(P)=2THENY(P)=Y(P)+1
100 IFD(P)=1THENX(P)=X(P)+1
110 IFD(P)=3THENX(P)=X(P)-1
120 IFX(P)<2ORX(P)>38THENE(P)=1
130 IFY(P)<1ORY(P)>19THENE(P)=1
140 IFVPEEK(X(P)+(Y(P)*40)+&H3C00)=229THENE(P)=1
150 VPOKE&H3C00+X(P)+(Y(P)*40),229
160 SOUNDP,Y(P)*30+110,11
170 IFX(1)=X(2)ANDY(1)=Y(2)THENE(1)=1:E(2)=1
180 NEXT
190 IFE(1)=0ANDE(2)=0THENGOTO50
200 GOTO1130
210 REM
220 IFSTICK(1)=1THEND(1)=0
230 IFSTICK(1)=5THEND(1)=2
240 IFSTICK(1)=7THEND(1)=3
250 IFSTICK(1)=3THEND(1)=1
260 IFSTICK(2)=1THEND(2)=0
270 IFSTICK(2)=5THEND(2)=2
280 IFSTICK(2)=7THEND(2)=3
290 IFSTICK(2)=3THEND(2)=1
300 RETURN
310 REM
320 A$=INKEY$
330 IFA$=CHR$(30)THEND(2)=0
340 IFA$=CHR$(31)THEND(2)=2
350 IFA$=CHR$(29)THEND(2)=3
360 IFA$=CHR$(28)THEND(2)=1
370 IFA$="W"THEND(1)=0
380 IFA$="Z"THEND(1)=2
390 IFA$="A"THEND(1)=3
400 IFA$="S"THEND(1)=1
410 RETURN
420 REM
430 GOSUB740
440 GOSUB1350
450 GOSUB770
460 GOSUB500
470 GOTO50
480 END
490 REM
500 SCREEN1,1:COLOR1,1:CLS
510 FORI=1TO18:PRINTCHR$(129);CHR$(131);:NEXT:PRINTCHR$(129);CHR$(136);
520 FORI=1TO9:FORA=1TO19:PRINT" ";CHR$(128);:NEXT:FORA=1TO18:PRINTCHR$(129);CHR$
(146);:NEXT:PRINTCHR$(129);CHR$(132);:NEXT:FORI=1TO19:PRINT" ":CHR$(128);:NEXT
530 FORI=1TO18:PRINTCHR$(129);CHR$(130);:NEXT:PRINTCHR$(129);CHR$(137);
540 FORA=&H3C01TO&H3F49STEP80
550 VPOKEA,133:VPOKEA+40,128
560 NEXT
570 VPOKE&H3F49,32
580 VPOKE&H3F21,135
590 VPOKE&H3C01,134
600 CURSOR14,21:PRINT"GRIDRUNNER"
610 CURSOR0,22:PRINT"SCORE (1):";SC(1)
```

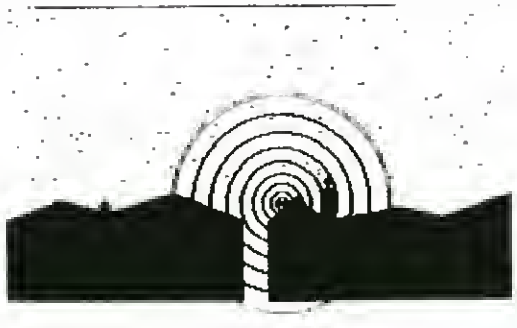
```
620 CURSOR25,22:PRINT"SCORE (2):";SC(2)
630 COLOR15,1
640 D(1)=2:D(2)=0
650 X(1)=2:Y(1)=0:X(2)=38:Y(2)=20
660 D=0:E=0
670 SOUND1,110,10
680 SOUND2,111,10
690 SOUND3,112,10
700 FORA=1TO500:NEXT
710 SOUND0
720 RETURN
730 REM
740 DIMSC(2),X(2),Y(2),D(2),E(2)
750 RETURN
760 REM
770 SCREEN1,1:COLOR15,1:CLS
780 PRINTTAB(14);"GRIDRUNNER"
790 PRINT:PRINT"Drive your gridrunner go-kart around"
800 PRINT"the matrix arena.  Avoid the tracks"
810 PRINT"of both you and your opponent's karts.";
820 PRINT"Avoid the walls, too, as they are also";
830 PRINT"instant death.  The last surviving"
840 PRINT"player is the WINNER!"
850 PPINT:PRINT"The controls are:"
860 PRINT"Key         Player 1          Player 2"
870 PRINT"UP            W                 ";CHR$(142)
880 PRINT"DOWN          Z                 ";CHR$(237)
890 PRINT"LEFT          A                 ";CHR$(143)
900 PRINT"RIGHT         S                 ";CHR$(238)
910 PRINT:PRINT"Alternatively joysticks may be used"
920 PRINT"with the stick in the JOY-1 socket"
930 PRINT"controlling player 1's kart and the"
940 PRINT"stick in the JOY-2 socket controlling"
950 PRINT"player 2's vehicle."
960 PRINT:PRINT"    Push the SPACE BAR to continue"
970 IFINKEY$<>" "THEN970
980 CLS
990 PRINT"Push a joystick button to select"
1000 PRINT"joysticks or push any key (within"
1010 PRINT"reason) to select keyboard."
1020 A$=INKEY$:J1=STRIG(1):J2=STRIG(2)
1030 IFA$<>""THENPRINT:PRINT"Keyboard selected":J=0:GOTO1060
1040 IFJ1<>0ORJ2<>0THENPRINT:PRINT"Joystick selected":J=1:GOTO1060
1050 GOTO1020
1060 PRINT:PRINT"Are you sure ? (Y/N)"
1070 A$=INKEY$
1080 IFA$="N"THEN980
1090 IFA$="Y"THEN1110
1100 GOTO1070
1110 CLS:RETURN
1120 REM
1130 SOUND0
1140 FORA=15TO0STEP-1
1150 SOUND4,2,A
1160 FORB=1TO20:NEXT
1170 NEXT
1180 SOUND0
1190 FORA=1TO300:NEXT
1200 CLS
1210 IFE(1)=1ANDE(2)=1THENCURSOR17,12:PRINT"DRAW":GOTO1260
1220 IFE(1)=1THENCURSOR11,12:PRINT"Player 2 wins!":GOTO1240
1230 IFE(2)=1THENCURSOR11,12:PRINT"Player 1 wins!":GOTO1240
1240 IFE(1)=1THENSC(2)=SC(2)+1
1250 IFE(2)=1THENSC(1)=SC(1)+1
1260 CURSOR9,15:PRINT"Another game ? (Y/N)"
1270 A$=INKEY$
1280 IFA$="N"THENCLS:PRINT"Bye....":END
1290 IFA$="Y"THEN1310
1300 GOTO1270
1310 E(1)=0:E(2)=0
1320 GOSUB500
1330 GOTO50
1340 REM
1350 SCREEN1,1:COLOR15,1:CLS
1360 CURSOR0,0
1370 PRINT:PRINT:PRINTSPC(14);"GRIDRUNNER"
```

```
1380 FORA=0TO36
1390 SOUND1,110+A*10,11
1400 CURSORA,9:PRINTCHR$(229)
1410 NEXT
1420 SOUND0
1430 PATTERNC#255,"3048B4A4A4B44830"
1440 PATTERNC#234,"FFFFFFFFFFFFFF00"
1450 PATTERNC#252,"00FFFFFFFFFFFFFF"
1460 PATTERNC#237,"20202020AB702000"
1470 PATTERNC#238,"001008FE0B100000"
1480 PATTERNC#143,"002040FE40200000"
1490 CURSOR9,12:PRINTCHR$(255);" N. Campbell 1985"
1500 CURSOR4,23:PRINT"Press the SPACE BAR to start";
1510 SOUND4,3,12
1520 FORA=4000TO200STEP-50
1530 SOUND3,A,0
1540 NEXT
1550 SOUND0
1560 IFINKEY$<>" "THEN1560
1570 RETURN
```

Graphics Screen Printer/ Plotter Dump Routine

Here is an interesting little program for those of you lucky enough to have a SEGA SP 400 Printer/Plotter. The program sent to us by G.W. Emms of Mairangi Bay in Auckland produces a graphics screen dump on your Printer. To do this the program examines the VIDEO RAM with the VPEEK on line 30. It breaks down the bytes of information into bits to see if there is a dot on the screen at that point. If the pixel is illuminated the computer tells the printer/plotter to produce a dot Notice that sprites are ignored as are changes in colour.

```
10 LPRINT CHR$(18);"H"
20 FOR Y=0 TO 191:FOR X=0 TO 255 STEP 8:I=INT(Y/8)*256+INT(X/8)*8+YMOD8
30 VP=VPEEK(I)
40 IF VP>=128 THEN LPRINT "J1,0":VP=VP-128:GOTO 60
50 LPRINT "R1,0"
60 IF VP>=64 THEN LPRINT "J1,0":VP=VP-64:GOTO 80
70 LPRINT "R1,0"
80 IF VP>=32 THEN LPRINT "J1,0":VP=VP-32:GOTO 100
90 LPRINT "R1,0"
100 IF VP>=16 THEN LPRINT "J1,0":VP=VP-16:GOTO 120
110 LPRINT "R1,0"
120 IF VP>=8 THEN LPRINT "J1,0":VP=VP-8:GOTO 140
130 LPRINT "R1,0"
140 IF VP>=4 THEN LPRINT "J1,0":VP=VP-4:GOTO 160
150 LPRINT "R1,0"
160 IF VP>=2 THEN LPRINT "J1,0":VP=VP-2:GOTO 180
170 LPRINT "R1,0"
180 IF VP>=1 THEN LPRINT "J1,0":GOTO 200
190 LPRINT "R1,0"
200 NEXTX
210 LPRINT "M0,";STR$(-Y):NEXT Y
```



Flip!

by Michael Howard

The following program is just a machine-code routine that is really quite silly! All it does is change the character set so they are all mirror images. To run just enter the program, run it, then type NEW. To actually run the machine code enter CALL&HFØØØ, a second call will return the character set to normal. The reason why the machine code is still around even though you typed new, is that the code is stored above &HFØØØ which is above a thing called RAMTOP. When you enter new the computer deletes all program up to RAMTOP, anything stored above RAMTOP is safe. Note: only machine code NOT basic can be stored above RAMTOP.

Flip!

```
1Ø DATA F3,DB,BF,21,Ø,18,1,Ø,8,C5,E5,CD,32,2C,DB,BE,E,Ø,6,6,17,CB,19,1Ø,FB,E1,E5,C5,CD,44,2C,C1,79,D-
3,BE,E1,C1,23,B,78,B1,2Ø,DE,C9.
2Ø RESTORE: FOR A=&HFØØØ TO &HFØ2B:READ A$:POKE A,VAL("&H"+A$):NEXT
```

Machine-code break-down

```
di          interrups disabled  } necessary
ina,BF      port BF read        }
ldhl,18ØØ   load hl register with starting address of characters in VRAM
ldbc,Ø8ØØ   load bc with number of bytes
push bc     save bc
push hl     save hl
call 2c32   } read in data at hl in VRAM, into a register
ina, BE     }
ldc,Ø       }
ldb,6       } does the dirty work! The c register will end up holding the new data. b is a
rla         } counter. a register is rotated left (see below), and this data is rotated into c.
rrc         } (If it sounds complicated — don't worry!)
djnz        }
pop hl      restore hl, remember hl holds an address
push hl     re-save hl
push bc     save new data (remember c reg. holds new data)
call 2c44   }
pop bc      } Out put new data, by placing in the same area as old data. (Call ROM
lda,c       } routine to point to new address in hl. Restor bc, output it to BE
out BE,a    }
pop hl      restore hl — address pointer
pop bc      restore bc — counter
inc hl      increase hl. (hl=hl+1)
dec bc      decrement bc. (bc=bc-1)
ida,b       }
orc         } check if bc=Ø, if not jump back if it is return to BASIC.
jmz         }
ret
```

An explanation of how RLA and RRC work. If you don't understand the concept's of binary arithmetic then I recommend you purchase "Teach Yourself Basic Game Programming" released by Grandstand.

Imagine you have a number say 44 and this is stored in Register "a" (a register is the machine code equivalent of a variable, and I reckon it sounds better than variable!) Now in binary 44 is: ØØ1Ø11ØØ. So in this case "a" holds ØØ1Ø11ØØ, okay!?

Just diverging a little, a very important "Function" in machine code is a little thing called "carry". Basically, carry is always set at "Ø", but if some operation takes place, like a register holds 255 (11111111) and it is increased by 1, there is an "overflow" this is because an 8-bit byte can only hold 8 bits! and by adding 1 would make 256 which is 9 bits (1ØØØØØØØØ), the 1 at the front of the number goes into the carry bit ("1") and the other 8 bits go to the register in question.

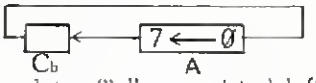
Here is what I mean. Carry = Ø, register = 11111111

```
register = 11111111
plus 1   +        1
         ----------
         1ØØØØØØØØ   carry = 1, register = ØØØØØØØØ
Carry ↗   register
```

Understand? Good! If you don't just re-read the above and think about it!

cont. from p7

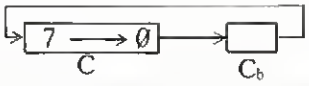
Now that you know how the carry bit works let me explain RLA. Remember that the register contains ØØ1Ø11ØØ. The RLA command looks like this:



A — "a" register
C_b — carry bit
7,Ø — bits 7,6,5,4,3,2,1,Ø in "a" ,

RLA — Rotate accumulator ("a" — register) left through carry flag.

RRC looks like



C — "c" register
C_b — carry bit
7,Ø — bit 7,6,5,4,3,2,1,Ø in "c" register

RRC — rotate right "c" — register through carry flag.

Now that we know how RLA,RRC and the carry bit works we can get down to how the reversing works. In our example "a" contains ØØ1Ø11ØØ "c" contains ØØØØØØØØ, carry bit is Ø. Okay look at the following break-down:-

Step	a	c	Carry
1	ØØ1Ø11ØØ	ØØØØØØØØ	Ø
2	Ø1Ø11ØØØ	ØØØØØØØØ	Ø
3	1Ø11ØØØØ	ØØØØØØØØ	Ø
4	Ø11ØØØØØ	1ØØØØØØØ	1
5	11ØØØØØØ	Ø1ØØØØØØ	Ø
6	1ØØØØØØØ	1Ø1ØØØØØ	1
7	ØØØØØØØØ	11Ø1ØØØØ	1
8	ØØØØØØØØ	Ø11Ø1ØØØ	Ø
9	ØØØØØØØØ	ØØ11Ø1ØØ	Ø

Can you see what is happening? "a" is being shifted left, bit 7 is put into the carry bit, now "c" is shifted right and the contents of the carry bit are placed in bit 7 of "c", this is repeated until "c" is the mirror image of "a"... voila!

You see its quite easy really, but if don't understand it, don't loose sleep over it, it's not really of political importance, unless you want to get right into machine-code!

---

cont. from p3

the aerodynamics of a mitsubishi on the banking at the side of the road. None the worse for wear our two day stint in Wellington gave us the opportunity to meet more dealers and promote our forthcoming T.V., and magazine campaign.

Saturday evening saw our intrepid team, (train to follow) sat dockside looking ruefully out to see, regretting that last sausage roll at lunchtime, listening to each others previous experiences on pitchind decks or overfull bathtubs. The actual journey to Picton was quite an anticlimax as we played and undisturbed game of trivial pursuits with barely a ripple on the water. The only disturbance infact came from a lone American, on an extended vacation who we befriended to prevent him ruining our game by giving all the answers and making us feel more stupid than was good for our morale.

Sunday morning presented the opportunity for another team member, who shall also be nameless to challenge for the driver of the year award. This time the infortunate mitsubishi, ran into trouble in a car park as an unnoticed boulder collided with the rear end of the vehicle, which came to rest with the back end suspended in mid air, and the back axle firmly wedged on top of the offending rock. One hour and one hydraulic jack later, the team were en route to Christchurch.

Situated at the top of Manchester Street, on platform one of Christchurch Station, the Sega/Amstrad express, made a real impression on the public at large. The computers once again were well received, and our T shirts did not go unnoticed either. Our American friend once again emerged to become nicknamed "double vodka tonic" and the town became a popular and enjoyable venue for all.

The final to the trip was Dunedin, and the long journey south was well worth the trip. Ahead of us lay the long ride home, so any thought of celebrations were tempered by the tought of a 2,000km drive home the next day.

This we completed with no further mishap, sept the unusual disappearance of an aluminium windscreen trim which put the finishing touch to the impression that all the cars had actually participated in the Monte Carlo Rally rather than a trade launch.

The whole journey left us all much the wiser and very pleased with our efforts. We now feel confident in the knowledge that both Sega and Amstrad owners will be able to buy product across the country from a network of dealers who can give the best support and advice possible in every area, and help keep us right at the very top of the tree in 1985 in computers.

BIG CHARACTERS PROGRAM

This program produces characters on the graphics screen twice as high and wide as they normally are. The computer looks at the information located at &H10C0 which is the hexidecimal data for the shape of the character patterns. Just type in any thing you want printed the larger script. This is a very useful little routine which can be incorporated into large programs.

memory cartridge only

```
10 DEF FN C(X)=&H10C0+((X-32)*8)
20 SCREEN 2,2:CLS:PATTERN C#255,"C0C0":Y1=10:X1=10
30 A$=INKEY$:IF A$<" "THEN 30
40 BEEP
50 X=ASC(A$):X=FN C(X):Z=X1
60 FOR A=X TO X+7
70 Z=Z+2:P=PEEK(A):RESTORE
80 FOR B=0 TO 7
90 READ D
100 IF D AND P THEN CURSOR Y1+B*2,Z:PRINT CHR$(255)
110 NEXT B,A:Y1=Y1+15:IF Y1>240 THEN Y1=10:X1=X1+16
120 GOTO 30
130 DATA 128,64,32,16,8,4,2,1
```

OVERSEAS REVIEW

BY PHIL KENYON

Having returned from my recent world tour to find myself toppled from the front pages, you can rest assured that you cannot escape my insane ramblings completely. The magazine just isn't the same without at least one page from my flowing nib, with this in mind the following article is a review of the first leg of the trip, in an attempt to keep you all up to date in the world of consumer electronics.

Hong Kong

Renowned for its phenomenal manufacturing capability, one might expect Hong Kong to be positively overflowing with computers and peripherals in almost every electrical shop. Anyone visiting with this hope in mind is in for a big disappointment.

The strength of the manufacturers is of course in their ability to copy and reduce the price on existing machines, which works fine if it's a clock, radio or a calculator, but when it comes to computers you run into problems with the law of copyright, which means that the huge proliferation of Apple and IBM copies, which Hong Kong manufacturers have concentrated on producing are all hidden away under the counter, from where they are brought with alarming ease, and regularity.

The European and American home computers were almost completely unrepresented in the stores, the language problem being one obvious drawback, but the universal attraction of the games playing capability of computers would arguably be a sufficient attraction to a big enough percentage of the population to make it an attractive market.

Sadly for Hong Kong, their currency has fallen against the Yen just as much as everyone elses, causing a dramatic increase in the price of photographic equipment, Japanese T.V. and Audio, and inevitably the new crop of MSX Micro's which continue to show promise but very little else, as they too are as scarce as hens teeth in the electronics shop front.

In the U.K. it is a slightly different story as the High Street computer shops are now displaying a variety of MSX computers. The comparatively high cost of the systems by comparison to Amstrad, Commodore, and Sinclair means that they have an uphill battle on their hands to sell the system.

The MSX software shortage is not being aided by the international software houses so far ignoring the system, apparently due to an unwillingness on the part of the manufacturers to give any sort of incentive, or up-front development payment to encourage confidence in the saleability of the system.

This in turn has led to the majors such as Boots, Argos and W H Smiths, being cautious and not introducing them to their range, which is undoubtedly what must happen for any machine to survive. This chain must be broken or I fear all of the MSX machines will fall by the wayside in what is sure to be a monumental battle in 1985 for supremacy in a huge and expanding UK market.

Sega, although not yet released in Britain is remarkably well known due to the occasional reviews in UK magazines eager for new material. "Big in Japan" is a common label as not many realise its popularity in Europe, Australia and New Zealand. Two of the major UK software houses I visited are considering introducing Sega Hardware to expand their operations, which would undoubtedly be good news for Sega owners everywhere.

The big news amongst the trade in England is the second major failure of a UK company to succeed in the lucrative but volatile US market, as BBC closed down its short lived American operations. BBC sustained huge losses trying to launch Acorn and suffered the same fate as the Timex Sinclair operation, who also tried to move in on the Americans, without success.

In their own home market the two giants of BBC and Sinclair continue to slug it out, and not only in the High Street. The battle has now spilled over into the bar-room, as the two computer chiefs, Sir Clive Sinclair and head of BBC Chris Curry exchanged blows and insults in the Baron of Beef pub in Cambridge, Surry.

Sir Clive reportedly started the incident by being angered by an Acorn advert, which referred to Sinclair products in an uncomplimentary way, and by beating Mr Curry about the head with a copy of the advert. According to "The Daily Mail" Mr Curry retaliated by punching his opponent.

Not to miss such an opportunity on English software house writing for the rival Amstrad, immediately set to work writing a program called "This Business is now War" based on the infamous brawl, but throwing computers instead of punches.

Prices in general have now stabilized with most machines selling for much the same as last year. Software trends are moving towards the more sophisticated entertainment programs which have been labelled "Action Strategy" combining greater mental agility with reaction in ingeneous situations. More and more people are also being introduced to the business applications of small micro's and realising their power in this environment with excellent low cost financial packages, which are continuing to underline more practical uses of computers.

The growing acceptance of computers in the home, reflecting by the success of large department stores in selling them, seems to have eased the nerves of most of the major suppliers. The computer business still remains a fashion industry and as such manufacturers must continue to stay alert to the ever changing trends. The one unanimous factor in which they can all take comfort, is that no-one can dispute the computer, in one form or another, is here to stay.

ROMAN NUMERALS

Not By Michael Howard!

"Friends, kiwis, country men, lend me your computer, I've come to show you a neat program, not to put it down" — Famous quote from a looney programmer!

Now I know, that you've all seen programs for converting Binary to Decimal, Decimal to Hex, Hex to Binary etc. Well this little program allows you to enter decimal and numbers and get Roman numbers out (eg 19 = IXX). Or you can enter Roman numbers and get decimal ones! (eg VII = 7). The only restrictions are that when entering decimal numbers you can only enter those from 0-4999, and in Roman numerals I-(4999 in Roman (whatever that is!))

The reason why the program is so long is that in the Roman numbers system, position of character is not significant. Let me expalin.

The number 17 is made up of 10 and 7.

```
  10
+  7
  17
```

Now 1 is called the most significent digit and 7 the least significent. But in Roman numeralogy IV (which is 4) shows no significance (like the pun? No! Oh!) to the above example, V is more significent than I, which is stupid! So this program has to work out which is more the important digit!

To use the program just RUN it, if you want to enter a decimal number just enter it, if a Roman numeral is to be entered just enter it! The program work's out what you want automatically!

I hope you like it!

```
10 R$="IXCMVLD"
20 X$=""
30 INPUT "ENTER NUMBER ";A$
40 IF A$>"A" THEN 180
50 FOR J=1 TO LEN(A$)
60 Y$="":A=VAL(MID$(A$,LEN(A$)-J+1,1))
70 IF A AND A<5 OR A=9 THEN Y$=Y$+MID$(R$,J,1)
80 IF A>3 THEN 140
90 A=A-1
100 IF A>0 THEN 70
110 X$=Y$+X$
120 NEXT J
130 PRINT X$:GOTO 340
140 IF A=9 THEN 170
150 IF J=4 THEN 90
160 Y$=Y$+MID$(R$,J+4,1):A=A-5:GOTO 100
170 Y$=Y$+MID$(R$,J+1,1):GOTO 110
180 IF A$<"C" THEN 340
190 T=0
200 FOR J=2 TO LEN(A$)
210 G=J-1:GOSUB 320:B=I
220 G=J:GOSUB 320:E=I
230 IF E>B THEN 270
240 T=T+B
250 NEXT J
260 G=LEN(A$):GOSUB 320:T=I+T:PRINT T:GOTO 340
270 P=J-1
280 IF MID$(A$,P,1)<>MID$(A$,J-1,1) THEN 240
290 G=P:GOSUB 320:E=I:T=T-2*E
300 P=P-1:IFP>=1 THEN 280
310 GOTO 240
320 RESTORE:FOR X=1 TO 7 READ I$,I:IF I$=MID$(A$,G,1) THEN RETURN
330 NEXT X
340 END
350 DATA I,1,V,5,X,10,L,50,C,100,D,500,M,1000
```

LEO LION

by GRANDSTAND

This graphic demonstration makes use of the SEGA's 32 sprite planes as well as the high resolution graphics.

Line 40-50:
These two lines read in information from lines 100-160. The computer defines character number 255 as the shape read in from the data lines. This is now printed on the screen in red. As a pixel remains turned on until it is turned off on the graphics screen we can now redefine character 255 and print the new shape on the screen.

Line 60:
This reads in the sprite data from lines 170-250.

Line 70:
This reads the X and Y coordinates and the colour of the sprites defined in line 60 and positions them on the screen.

Line 80-90:
This is the clever bit that causes the lion to wink. Line 80 redefines the sprite making the lions eyes, nose and mouth. Line 90 makes the "roar" and changes the sprite back to what it was before.

```
10 REM *** Leo Lion by GRANDSTAND ***
20 SCREEN 2,2:CLS:RESTORE 100
30 COLOR 6,15,(0,0)-(255,191),15
40 FOR I=1 TO 37
50 READ X,Y,A$:PATTERN C#255,A$:CURSOR X,Y:PRINTCHR$(255):NEXT
60 FOR I=0 TO 35:READ A$:PATTERN S#I,A$:NEXT
70 MAG 1:FOR I=0 TO 8:READ X,Y,C:SPRITE I,(X,Y),I*4,C:NEXT
80 FOR DE=1 TO 200:NEXT DE:PATTERN S#12,"606000000001F4F07":PATTERN S#13,"52079F7
707070200":PATTERN S#14,"C0C0000000C09000"
90 SOUND 3,1000,0:FOR DE=0 TO 15:SOUND 3,600-5*DE,0:SOUND 5,3,15-DE/3:NEXT DE:SO
UND 0:RESTORE 200:FOR I=12 TO 15:READ A$:PATTERN S#I,A$:NEXT:GOTO 80
100 DATA 50,50,0000000000000804,58,50,0000000001030303,66,50,071F3FFFFF833B7F,74
,50,E3F7FFFFFFFC180,82,50,F0FCFEFFFFE0EEFF,90,50,00000080C0E06060
110 DATA 50,58,0404060300000000,58,58,0303038163676767,66,58,73660C8808080808,74
,58,0041E3E3E3000008,82,58,6733908888080808,90,58,606060C060707070
120 DATA 50,66,0000000000000001,58,66,6767676767C3C181,66,66,080C0E048484E4E2,74
,66,081C3E0000000000,82,66,0818381010101323,90,66,70707070F0E0C0C0
130 DATA 50,74,0101030303060606,58,74,8181010100000000,66,74,E0E0F8F8F87E7EFF,74
,74,003E3E1C00000080,02,74,03030F0F0F3F3EFE,90,74,C0C0C08000000000
140 DATA 50,82,0606060603030000,58,82,0101030206FEFA03,66,82,BF171717131808080,74
,82,A2E3FFFFFF7F7F3E,82,82,FEF4F4F4E40C0808
150 DATA 58,90,0100000000030404,66,90,8CC4643612F20E10,74,90,3E1C1C1C1C1C1C1C,82
,90,1810103020203804
160 DATA 58,98,0507000000000000,66,98,50F51F0000000000,74,98,1C5DE30000000000,82
,98,0454FC0000000000
170 DATA 7CC4808C99F377F7,F6F6F6F7F3F1F878,003E7FFFBE1C1C1C,1C1C14F7E3C10000
180 DATA 1F110098CC677777,373737F7E7C70F0F,0080808080800080,8080808080808000
190 DATA 03060E0E1EF8F8F8,F0F0C0400404040C,8000000000000000,0000000000000000
200 DATA 1818000001F4F07,52079F7000000000,3030000000C09000,5000C87000000000
210 DATA 0000000000000000,00FE07030303010C,0000000000000000,000000080C0C0C040
220 DATA F1F1F10000000000,0000000000000000,E0E0E0000000000,0000000000000000
230 DATA 3C0C0C0E0F030303,0000000207474F4F6,0000000000E0F1FF,FFFF3F2F0E000000
240 DATA F3FB7B391D0D0406,0600000000000000,C0C0E0E0F0F0F0F0,E0E0000000000000
250 DATA 7878F8F0F0F0E0E0,E0E0000000000000,0000000000000000,0000000000000000
260 DATA 66,54,10,82,54,10,81,70,10,72,63,1,50,45,1,73,62,12,65,70,10,65,86,10,8
1,86,10
270 GOTO 270
```

SEGA HALL OF FAME

The person recording the highest points tally on each of the Sega games cartridges will receive a free cassette program, and have their name published in the Magazine, along with their score.

To have your score entered, we must have some form of verification as to its validity, ideally a photograph of the screen, showing the recorded high score. In games where the high score may be flashing, the screen action can be frozen, by depressing the Reset key.

Monaco GP	999,999	Steve Winter
Video Flipper	699,160	David Palmer
Starjacker	750,454	David Palmer
Exerion	587,400	David Palmer
Borderline	536,380	David Palmer
Champion Golf	-8	David Palmer
	-8	Hank Hoitanga
Pacar	3,246,000	Darryl Gore
Pop Flamer	226,000	Mark Cutler
Sinbad Mystery	302,050	Craig Woodward

REVIEWS

Here it is — Over 50 Programs for the SC-3000

by Michael Howard

The loony who brought you "The House!" and "Help!" has struck again! This time his skills are turned to book writing. But needless to say Purple People Eaters and Vanessa the Vampire raise their heads! Mike also brings us a new entity — Kamikaze Combat Caterpillars! and Mutant Apples (straight-jacket time!)

The book contains 56 programs ranging from an excellent conversion of the popular game "Simon" to a couple of complete adventure games, to a mob of graphics programs (including how to create 3D effects) to a load of spacies games. For the more serious users . . . complete graphics statistics packages, a grid sprite editor, quadratic regression, machine coded sound and scrolling, a few artificial intelligence programs, music programs, a morse code tester, a wierd version of "Star Trek" — (I can guarantee you'll never have seen one like this!), prime number, wave phase changes, also a chance to rule an empire . . . in short HEAPS of programmers delights! The list above is just a minute chip off the iceberg!

All the program's are explained fully, thus enabling you to learn how they work, and allowing you to modify them.

This book will see a winner and is well worth its money. Oh, by the way, the author also designed the cover. It shows a picture of Vanessa's offspring (called Lilvampet) reading a copy of the book. book.

But, seriously folks, the author should be locked up in a padded cell, the key thrown away, and all those who know of his whereabouts — shot!

After you've read a few of the stories accompanying the programs you'll see why!

Help! by Michael Howard

You are out prospecting for Televisions when suddenly you are nabbed by a gang of Purple entities, and sentenced by their leader "Throggle" to walk the lengths of the "Crazy Crypt". So goes the blurb. Those in the know will know who the Purple entities are . . . They are Purple People Eaters.

Apparently, when "The House" was desecrated by various adventurers, all called Siddy Superspook, the PPE's as a race were nearly obliterated. A few, however, under the leadership of "the Boss of all those Purple and Slimy" — Throggle fled The House, and took up residence in a posher part of town called "The Crazy Crypt". Here they lived and bred.

But how did they eat? Well people like you who decide to go on expeditions looking for fellies (TV) end up getting gobbled at dinner tables throughout the Crazy Crypt.

Okay, now I've set a morbid scene, it's time to describe it.

The program is described as a game with educational overtones, a game it certainly is, educational it most defininately is.

The object of the game, is to get from one side of the crypt to the other. Sounds easy! It's not! On the way various hazards are thrown your way. These range from an arcade-machine-code racing game in which you drive Ross's Anglia, to any of 4 general knowledge quizzes, or from meeting Charlie (who's got a size 13 feet!) to Revenge of the Purple People Eaters! This is just a fraction of the nasties out to nobble you! There is also Mick, the sorry programmer, Annie, the Hail of arrows, Dastardly Dick, Bill and more! In my mind, the worst you can meet is Merlin the Black Cat, but I'll let you discover why! One word of advice . . . stay on the path!

The very last bit of blurb says, "Everytime you die put 10 cents away . . . in a few months you'll be a millionaire!" . . . I believe it!

All-in-new an excellent program containing addiction (and frustration), for the games players and education (maths, keyboard skills, vocabulary, geography and books etc) for educationalists and parents. From the programmer of "The House" if you havn't guessed already!

Sprite Generator

If you are planning on writing your own game then this is definately recommended. Unlike the Sprite Editor this one allows you to draw your sprite on a grid. After drawing your sprite you can reflect and rotate your sprite.

You can even get a picture of your sprite printed out on the SEGA printer/plotter along with hexadecimal code of the sprite. Another option is to have your sprite codes put into a program for you which saves a lot of typing.

You can recall any sprite from another programme and edit it. To see your sprite in action you can move it around the screen. The sprite generator will save budding games programmers hours of keyboard punching.

Cycletron

Imagine roaring along on the latest laserbike, able to turn on 90° angles leaving a trail of radioactive laser dust in your wake. Well now you can with Grandstands latest piece of software, A two player game that involves both players riding space age laserbikes. The object of the game is to block your opponent in with a laser trail which is left behind as you roar around the screen. As players progress more pieces of space junk appear to hinder your progress. Once one rider is smashed against a laser trail the other must carry on and last as long as possible. A challenging and enjoyable game for the whole family.

World Geography – disk only

This is computerised geography at its best. This program tests the following 10 different geographical areas:

North Island New Zealand
South Island New Zealand
Australia
Pacific Region
North America
South America
British Isles
Europe
Japan and Korea
Africa

World Geography knows so many different locations and geographical features such as mountains, lakes and rivers that it is nearly impossible to memorise where everything is! The program makes learning geography fun and its maps clearly show where all the major feature of an area are located. World geography is made up of geography 1, 2 and 3 which are available on cassette.

Cosmic Combat disk and cassette

'Cosmic Combat' is a very peaceful game for two players. The scene is the distance reaches of the galaxy where, by coincidence, no man has gone before. In this particular part of the galaxy there happens to be a planet which is open to domination by passers by. As it happens two ships suddenly appear and obviously the only way to settle the ownership question is with a duel — using the aforementioned spaceships, along with their blasters.

In contrast to most computer games, this is a player vs player games rather than a player vs computer game, and as such can be a lot more fun than plugging away at a mechanical opponent for hours.

The goal of this game is to destroy your opponent 7 times. After this the winner takes possession of the planet as the loser fades into space.

This game is written extensively in machine code and so is very fast at the fastest selectable speed — 'slow' gives a tactical game whereas 'fast' introduces an element of luck.

Each player controls a ship using rotate left, rotate right, fire, in his/her fight to the finish.

Together, the speed, graphics (especially the explosions), and sound make this a program that everyone, especially those who would like to blast someone else out of the sky, should have.

Disk Software

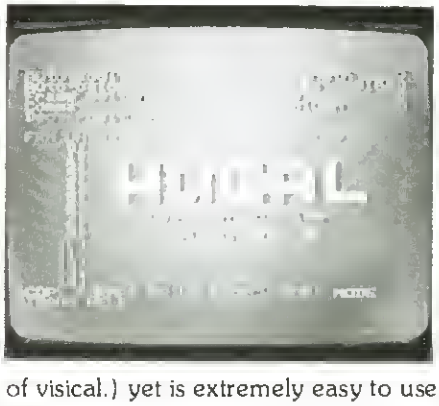
See last months issue for a tall review of this package.

The Sega SF7000 Super Control Station was released in New Zealand in December. This unit allows rapid storage and retrival of information (programs, files and other data) thus your Sega can be used more efficiently for filing systems and small business applications. The following education, entertainment and business programs are now available.

NOTE: HuCAL, LOGO and SEGA BASE have been ordered from Japan. Grandstand is awaiting shipping details so it is not possible to give a data for the release of these programs.

HuCAL – disk only

This is a very comprehensive computerised spread sheet, (Along the lines



of visical.) yet is extremely easy to use and understand. Its is capable of producing linked bar graphs, complex balance sheets, cashflow analysis, projecting costs etc. This package includes the following facilities and features (and many more!)

- list information to a parallel/series printer or to the Sega SP400
- Equipped with text windows
- 14 digit high accuracy calculations
- The number of decimal places and rounding can be specified for each column
- Automatic insertion of commas into numerical information (eg 123456789 becomes 123,456,789) This can be specified for each column
- Full on screen text editing
- Data search provided
- Data sort provided
- Macro commands including summation of rows and columns, and all standard mathematically functions (eg SIN, COS etc)
- Data protection of each column and row possible
- The spread sheet can be up to a massive 255 cells (horizontally) by 10001 cells (vertically) depending on the amount of vacant memory space
- Automatic calculation of row and column totals and Macro commands

This is a very professional business package that comes with a 175 page manual. HuCAL opens the door for serious business applications and cannot be ignored by anybody using their Sega in this area.

Logo – disk only

Welcome to the mysterious world of LOGO. (For those of you who are unfamiliar with LOGO it is a simple language used to teach people (especially children) how to use, understand and program computers)

This package includes full Turtle graphics, varibles, primitives or procedures and mathematically capabilities. There are TWO 150 page manuals supplied with the package. The first is a simple clearly written instruction manual for people who have not used LOGO before and the second is a very comprehensive run down of the more complicated features of LOGO. Both manuals are filled with working examples to help demonstrate the texts.

LOGO is a very powerful language as you can actually increase the artificial intelligence of your computer. It has been specially designed to perform graphs programming with the greatest of ease and can be mastered by anybody from 2 to 92.

Munchman

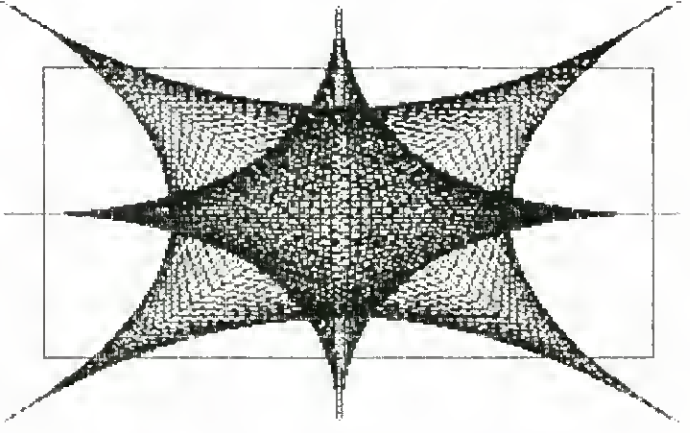
– disk and cassette

The most popular cassette based game for the SC3000 is now available on disk. Manouver your little dot munchin' man area the screen, avoiding the deadly ghost and keeping an eye out for the delicous fruit which appears in the maze. You only have four Power Pills to use against the ghosts so use them with the greatest of care. A good addictive arcade game that has caused many fingers to blister and callous.

Sega Base – disk only

A disk based Data base that will store up to 55K of information in one file! You can store details on everything from mailing lists to your stamp collection. Sega base will hold around 700-800 "normal" and addresses, allowing you to sort and select information from your file on the basis of various criteria. (string searchs, characters, numerical order etc) All the information may be listed on a parallel/series printer or on the Sega SP400 printer/plotter. A detailed manual accompanies this database program.

STRINGS



This program by Terry Johnson of Kaikohe won't provide you with The Meaning of Life nor will it conform The Theory of Relitivity, but it does look good!

```
10 SCREEN2,2:COLOR1,15:CLS
20 COLOR1,6,(16,30)-(245,161),15
30 FORX=0TO128STEP4
40 LINE(X,((191/255)*X))-(128-X,(191/255)*X+96),1
50 NEXTX
60 FORX=0TO128STEP4
70 LINE(255-X,(191/255)*X)-(127+X,(191/255)*X+96),1
80 NEXTX
90 FORX=4TO127STEP4
100 LINE(X,(96/127)*X)-(127+X,96-(96/127)*X),1
110 NEXTX
120 FORX=4TO127STEP4
130 LINE(X,191-(96/127)*X)-(127+X,(96/127)*X+96),1
140 NEXTX
150 FORX=0TO127STEP4
160 LINE(X,96)-(127,96+(96/127)*X),1
170 NEXTX
180 FORX=0TO127STEP4
190 LINE(X,96)-(127,96-(96/127)*X),1
200 NEXTX
210 FORX=0TO127STEP4
220 LINE(255-X,96)-(127,96-(96/127)*X),1
230 NEXTX
240 FORX=0TO127STEP4
250 LINE(255-X,96)-(127,96+(96/127)*X),1
260 NEXTX
270 FORX=1TO800:NEXTX
```

MEMORY MAPPING

The following pages are a small excert from Grandstands latest book that should be available in six months time. The book will be called the Concise Sega Firmware Specifications and will be about 200 pages long. It will cover

- The location and function of all the ROM and RAM routines
- Input registers for each routine
- Output information for each routine
- Complete memory maps for both the Disk Basic ROM (Version 1.0p) and the cartridge Basic ROM (Level 3 Version 1.0)
- Complete listings of the RAM system variables for both Basic ROMs

The manual will also cover other information on the machine code of the Sega.

NOTE: Due to the complexity of this manual it will not be available for another SIX MONTHS

CARTRIDGE BASIC ROM (Level 3 ver 1.0)

(Label)	(Address) hexadecimal	(Byte No.) decimal	(Contents)
QRAC	8000	8	Floating-point accumulator
QRST	8008	1	Floating-point status register
QROVS	8009	1	Sign at overflow time
QROEF	800A	1	Error processing table Refer to Chap. 1
QROEA	800B	2	
QRUEF	800D	1	
QRUEA	800E	2	
QRZEF	8010	1	
QRZEA	8011	2	
QRDEF	8013	1	
QRDEA	8014	2	
TXTBGN	8160	2	Beginning address of text storage area
ARYBGN	8162	2	Beginning address of subscripted variable storage area
VARBGN	8164	2	Beginning address of variable storage area
FREBGN	8166	2	Beginning address of free area
FREEND	8168	2	End address of free area + 1
LINEMD	816A	1	Internal code: Line mode
LINELN	816B	1	Line length − 1
LINENO	816C	2	Line No.
LINEIM	8170	256	Line image
PUTADR	82A0	2	PUT routine entry address
			Character string register 1:
STLEN1	82A2	1	Character length
STREG1	82A3	255	Contents of character string
			Character string register 2:
STLEN2	83A2	1	Character length
STREG2	83A3	255	Contents of character string
ASTACK	84A2	256	Numeric operation stack
ASTPTR	85A3	2	Stack pointer
BSTACK	85A5	320	Character string operation stack
BSTPTR	86E6	2	Stack pointer
STACKB	8730	1024	CPU stack area
RNDGEA	8B30	1	Species of random number
RNDGEB	8B31	1	Species of random number
RNDGEC	8B32	1	Species of random number
ROUNDB	8B34	1	Type of rounding
ROUNDC	8B35	1	Point of rounding
VRAMSV	8B36	2048	VRAM save area
VRAMMD	9336	1	Display mode
SPRSIZ	9337	1	Sprite size
SPRMAG	9338	1	Sprite magnification
COLORT	9339	1	Text screen color
COLOTG	933A	1	Graphics screen color
COLORD	9410	1	Back drop color
SCRULM	9411	1	Text screen upper limit
SCRLLM	9412	1	Text screen lower limit
PATERN	9413	8	Pattern buffer
KEYIMO	9460	1	Key input buffer (raw data)
KEYIM1	9461	1	
KEYIM2	9462	1	
KEYIM3	9463	1	
KEYIM4	9464	1	
KEYIM5	9465	1	

KEYIM6	9466	1	
KEYIM7	9467	1	
KEYIMC	9468	1	
KEYIR0	9473	1	
KEYIR1	9474	1	
KEYIR2	9475	1	
KEYIR3	9476	1	Key input buffer
KEYIR4	9477	1	(rising data)
KEYIR5	9478	1	
KEYIR6	9479	1	
KEYIR7	947A	1	
KEYIA0	947B	1	
KEYIA1	947C	1	
KEYIA2	947D	1	
KEYIA3	947E	1	Key input buffer
KEYIA4	947F	1	(auto repeat data)
KEYIA5	9480	1	
KEYIA6	9481	1	
KEYIA7	9482	1	
GREENG	9484	1	Switching among alphabet/dieresis/graphics
SMLCAP	9485	1	Switching between uppercase/lowercase
KEYCRC	9486	1	Presence/absence of click
CURCOL	9489	1	Cursor, column
CURROW	948A	1	Cursor, row
TIMEA	948B	2	Time adjustment counter
TIMEI	948D	1	(1/60,1/50)
TIMES	948E	1	Second
TIMEM	948F	1	Minute
TIMEH	9490	1	Hour
PRTADR	94A0	2	Printer output routine entry
CRTADR	94A2	2	Display output routine entry
KEYBUF	94A4	257	Key input buffer
CSTACK	95A7	112	GOSUB stack
CSTPTR	9618	2	Stack pointer
DSTACK	961A	200	FOR/NEXT stack
DSTPTR	96E3	2	Stack pointer
CHSIZE	96FE	1	Graphics screen:Character size
GRCURX	96FF	1	Cursor X
GRCURY	9700	1	Cursor Y
CRTMOD	9701	1	CRT mode
EXLINE	9705	2	Line number at execution time
EXADDR	9707	2	Address of next line
POWERF	97E2	1	Power ON flag

DISK BASIC ROM (ver 1.0p)

(Label)	(Address) hexadecimal	(Byte No.) decimal	(Contents)
QRAC	9800	8	Floating-point accumulator
QRST	9808	1	Floating-point status register
QROVS	9809	1	Sign at overflow time
QROEF	980A	1	
QROEA	980B	2	
QRUEF	980D	1	
QRUEA	980E	2	Error processing table
QRZEF	9810	1	Refer to Chap. 1
QRZEA	9811	2	
QRDEF	9813	1	
QRDEA	9814	2	
TXTBGN	9954	2	Beginning address of text storage area
ARYBGN	9956	2	Beginning address of subscripted variable storage area
VARBGN	9958	2	Beginning address of variable storage area
FREBGN	995A	2	Beginning address of free area
FREEND	995C	2	End address of free area and + 1
LINEMD	995E	1	Enternal code: Line mode
LINELN	995F	1	Line length − 1
LINENO	9960	2	Line No.
LINEIM	9964	256	Line image
PUTADR	9A8B	2	PUT routine entry address
GETADR	9A8D	2	GET routine entry address

			Character string register 1:
STLEN1	9A8F	1	Character length
STREN1	9A90	255	Contents of character string
STLEN2	9B8F	1	Character string register 2: Character length
STREG2	9B90	255	Contents of character string
ASTACK	9C8F	256	Numeric operation stack
ASTPTR	9D90	2	Stack pointer
BSTACK	9D92	320	Character string operation stack
BSTPTR	9FD3	2	Stack pointer
STACKB	9F10	1024	CPU stack area
RNDGEA	A310	1	Species of random number
RNDGEB	A311	1	Species of random number
RNDGEC	A312	1	Species of random number
ROUNDB	A314	1	Type of rounding
ROUNDC	A315	1	Point of rounding
VRAMSV	A316	2048	VRAM save area
VRAMMD	AB16	1	Display mode
SPRSIZ	AB17	1	Sprite size
SPRMAG	AB18	1	Sprite magnification
COLORT	AB19	1	Sprite screen color
COLORG	AB1A	1	Graphics screen color
COLORD	ABE1	1	Back drop color
SCRULM	ABE2	1	Text screen upper limit
SCRLLM	ABE3	1	Text screen lower limit
PATERN	ABE4	8	Pattern buffer
KEYIM0	AC1B	1	Key input buffer (raw data)
KEYIM1	AC1C	1	
KEYIM2	AC1D	1	
KEYIM3	AC1E	1	
KEYIM4	AC1F	1	
KEYIM5	AC20	1	
KEYIM6	AC21	1	
KEYIM7	AC22	1	
KEYIMC	AC23	1	
KEYIR0	AC2E	1	Key input buffer (rising data)
KEYIR1	AC2F	1	
KEYIR2	AC30	1	
KEYIR3	AC31	1	
KEYIR4	AC32	1	
KEYIR5	AC33	1	
KEYIR6	AC34	1	
KEYIR7	AC35	1	
KEYIA0	AC36	1	Key input buffer (auto repeat data)
KEYIA1	AC37	1	
KEYIA2	AC38	1	
KEYIA3	AC39	1	
KEYIA4	AC3A	1	
KEYIA5	AC3B	1	
KEYIA6	AC3C	1	
KEYIA7	AC3D	1	
GREENG	AC3F	1	Switching between alphabet/graphics
SMLCAP	AC40	1	Switching between uppercase/lowercase
KEYCRC	AC41	1	Presence/absence of click
CURCOL	AC44	1	Cursor, column
CURROW	AC45	1	Cursor, row
TIMEA	AC46	1	Time adjustment counter
TIMEI	AC48	2	(1/60, 50)
TIMES	AC49	1	Second
TIMEM	AC4A	1	Minute
TIMEH	AC4B	1	Hour
PRTADR	AC4C	2	Printer output routine entry
CRTADR	AC4E	2	Display output routine entry
DSKADR	AC50	2	Disk output routine entry
PSIOAD	AC52	2	RS-232C output routine entry
GKBDAD	AC54	2	Keyboard input routine entry
GDSKAD	AC56	2	Disk input routine entry
GSIOAD	AC58	2	RS-232C input routine entry
KEYBUF	AC5A	257	Key input buffer

CSTACK	AD5E	112	GOSUB stack
CSTPTR	ADCF	2	Stack pointer
DSTACK	ADD1	200	FOR/NEXT stack
DSTPTR	AE9A	2	Stack pointer
CHSIZE	AEB5	1	Graphics screen:Character size
GRCURX	AEB6	1	Cursor X
GRCURY	AEB7	1	Cursor Y
CRTMOD	AEB8	1	CRT mode
EXLINE	AEBD	2	Line No. at execution time
EXADDR	AEBF	2	Address of next line
POWERF	AF6D	1	Power ON flag
FILMAX	B02A	1	File max.
DIRBUF	B02B	256	Directory buffer
DIRPNT	B12B	1	Pointer
FATBUF	B12C	256	FAT buffer

cont. from p7

Again it is just the reverse of the above. The low byte of the register pair is loaded into the address following the opcode, this address is then incremented and the high byte is loaded into that address.

Register Indirect Addressing.

(When used as destination)

Let's say the computer is busy doing some calculations and then requires to go to a subroutine. This subroutine is going to change the contents of the registers it is now using, therefore it is going to have to save the contents of the registers into memory. The program will have to generate this memory space and not only that, it will have to remember were this memory space is loaded, when it returns from the subroutine. Now this would normally be a very difficult task if it was not for the Stack Pointer register. Now the Stack Pointer (SP) holds the sixteen bit address of a block of memory set aside especially for this purpose. Before we went to the subroutine the Stack Pointer will normally have been loaded with its address early in the program, let's say FFFF$_H$. To save the AF register pair we would load the opcode F5$_H$, the data from the low byte (F) will be copied into the memory location pointed to by the stack pointer which is FFFF$_H$. The stack pointer is now "Decremented" (FFFE$_H$) and the data from the high byte (A) is copied into that location. This would continue for all the registers we wished to save, it is at this time we would go to the subroutine.

When the subroutine was completed, we would use the opcodes to source the information back to the registers they had come from. REMEMBER that the stack is a Last in First Out file, so you must reverse the order in which you placed them on the stack.

e.g., If we placed or PUSHed the registers AF,BC,DE,HL we would have to return or POP them as HL,DE,BC,AF in that order.

It is in this way that we can load data from one register pair to another. If we 'PUSHed' reg BC we could use a different source code to put the data into another register pair.

ie if we wanted to load the contents of HL with the contents of BC we would PUCH BC followed by POP HL.

With the information I have given so far there is not a lot of programming you can do except for data movement. In the next issue we will be starting on ARITHMATIC and LOGIC, so we sould start to develop some useful programs then.

There was a bit missing from the program I gave you last issue so here it is:-

```
10 FOR X=&HA000 TO &HA007
20 READ A : POKE X,A : NEXT X
30 DATA &H16,&H60
40 DATA &H7A
50 DATA &H21,&H10,&HA0
60 DATA &H77
70 DATA &HC9
80 CALL&HA000
90 END
```

The answers to last issues assignment:-

1. 16 (0 to F)
2. 2=0010, 6=0110, 32=0010 0000, 11=1011, 10=1010, 15=1111, 8=1000, 14=1110
3. a=0001 0100, b=0000 0100, c=0001 0101, d=0111 1000
4. a=13+7=20, b=2+2=4, c=13+8=21, d=103+17=120
5. a=1111 0001 0100 1010
   b=0000 0101 0010 0001
   c=0100 1001 0000 0010
   d=1100 0000 1101 1110

Readers letters cont. from page 2

many men you have to rescue during each stage add these alterations:

```
990 CURSOR 200,64:PRINT MA:CURSOR 200,88:PRINT CH;" (":CURSOR 224,88:PRINT ST+11;")":CURSOR 200,112:PRINT WA
```

In line 1410 change (231,95) into (220,95)

Perhaps other readers will enjoy these small changes.

Andrew Ennever Massey Auckland

EDITORS REPLY

Good modifications. This makes the game a lot more challenging.

DEAR EDITOR

If you are to mention logarithums in the next issue of our club magazine, perhaps you should cover all 3 types available on the SEGA: base 10, base e and base 2.

Only base e is covered in the handbook. The complete listing is as follows:

```
log10X PRINT LGT(X)
logeX PRINT LOG(X)
log2X PRINT LTW(X)
```

Antilogs are obtained as follows:

Antilog of a number (base 10)

```
PRINT 10^X
```

Antilog of a number (base e)

```
PRINT EXP(X)
```

Antilog of a number (base 2)

```
PRINT 2^X
```

Dr A.S. Campbell
Canterbury

NEW SJ 300 JOYSTICKS ARE HERE!

New Sega Command Control Joysticks now available from your local Sega Dealers.

* LARGE CHUNKY CONTROL STICK.
* SEPARATE FIRE BUTTONS ON BASE.
* EXTREMELY ACCURATE AND RESPONSIVE.
* A MUST FOR ALL SERIOUS GAME PLAYERS.

PROGRAM OF THE MONTH

Just to show we're not just interested in playing games of spacies this months program is a calender utility program by Keith Surridge of Wainuiomata. This program uses no graphics and lacks action but is very interesting and useful.

Also anyone not in the Wellington province will need to substitute their own local holiday and delete Wellington's Anniversary Day.

```
10 CLS:PRINT :PRINT
20 PRINT TAB(9);"********************"
30 PRINT TAB(9);"*    CALENDAR      *"
40 PRINT TAB(9);"********************"
50 PRINT :PRINT : PRINT TAB(2);"A Perpetual Calendar, Day-of-Week, "
60 PRINT :PRINT :PRINT TAB(8);"and Holiday Programme"
70 PRINT :PRINT
80 PRINT TAB(10);"by KEITH SURRIDGE"
90 FOR K=0 TO 1500:NEXT
100 DIM N(12),M$(12)
110 FOR X=1 TO 12:READ M$(X):NEXT X
120 DATA JANUARY,FEBRUARY,MARCH,APRIL,MAY,JUNE,JULY,AUGUST,SEPTEMBER,OCTOBER,NOV
EMBER,DECEMBER
130 FOR X=0 TO 12:READ N(X):NEXT X
140 DATA 0,31,28,31,30,31,30,31,31,30,31,30,31
150 FOR X=1 TO 7:READ J$(X):NEXT X
160 DATA SUNDAY,MONDAY,TUESDAY,WEDNESDAY,THURSDAY,FRIDAY,SATURDAY
200 REM SELECTION ROUTINE
210 CLS:PRINT :PRINT
220 PRINT : PRINT "You can select a Calendar for any     Month":PRINT
230 PRINT "or the Day of the Week for any Date":PRINT
240 PRINT "or the Holidays in any Year"
250 PRINT:PRINT
260 PRINT "Select  CALENDAR     (C)"
270 PRINT "        DAY-OF-WEEK  (D)"
280 PRINT "        HOLIDAYS     (H)"
290 PRINT :PRINT :PRINT  "Type in C,D or H:-"
300 B$=INKEY$:IF B$="" THEN 300
310 IF B$="D" THEN 3000
320 IF B$="H" THEN 4000
330 IF B$<>"C" THEN PRINT "Invalid Entry":GOTO 290
340 GOTO 2000
400 CLS:PRINT :PRINT "  Calendar Programme says GOODBYE!"
410 END
1000 REM DAY OF WEEK CALCULATION
1010 K=INT(.6+1/M):L=Y-K:O=M+12*K:P=L/100
1020 Z1=INT(P/4):Z2=INT(P):Z3=INT(5*L/4):Z4=INT(13*(O+1)/5)
1030 Z=Z4+Z3-Z2+Z1+D-1
1040 Z=Z-(7*INT(Z/7))+1
1050 RETURN
1100 REM  EASTER ROUTINE
1105 IF Y<1900 OR Y>2099 THEN PRINT :PRINT "The Easter Calculation is only valid
  between 1900 and 2099.":GOTO 4270
1110 R=Y-1900:A=RMOD19:B=INT((7*A+1)/19).
1120 M=0:M=(11*A+4.00001-B)/29
1130 X=M-INT(M):IF X=1 THEN M=0
1140 IF X<>1 THEN M=29*X
1150 Q=INT(R/4):W=(R+Q+31-M)/7:W=7*(W-INT(W)):W=INT(W):DE=INT(25-M-W)
1160 IF DE>0 THEN M=4
1170 IF DE<0 THEN M=3
1180 IF DE=0 THEN M=3:D=31
1190 IF DE<-9 THEN DE=DE+9:GOTO 1190
1200 IF DE<0 THEN D=31-ABS(DE)
1210 IF DE>0 THEN D=DE
1220 IF Y=1974 OR Y=1984 THEN D=D+7
1230 IF Y=1994 THEN D=D-24:M=4
1240 RETURN
2000 REM CALENDAR ROUTINE
2010 FOR X=1 TO 50:NEXT X:CLS
2020 PRINT :INPUT "Year required ?";Y
2030 IF Y>1752 THEN 2050
```

```
2040 PRINT :PRINT "Year must be after 1752":GOTO 2020
2050 LY=0:IF(YMOD4=0 AND YMOD100<>0)OR(YMOD400=0)THEN LY=1
2060 PRINT :INPUT "Month (number not name) ?";M
2070 IF M<1ORM>12 THEN PRINT "Invalid Month!":PRINT :GOTO 2060
2080 D=1:GOSUB 1000
2090 N=N(M):IF M=2 AND LY=1 THEN N=N+1
2100 CLS:PRINT TAB(10);"CALENDAR FOR";Y
2110 Q=19-(LEN(M$(M))/2):PRINT :PRINT TAB(Q);M$(M)
2120 PRINT "************************************"
2130 PRINT "   S    M    T    W    T    F    S"
2140 PRINT "************************************"
2150 Q=2+5*(Z-1)
2160 PRINT TAB(Q);1;
2170 FOR G=2 TO 9:Q=Q+5
2180 IF Q>33 THEN PRINT :PRINT :Q=Q-35
2190 PRINT TAB(Q);G;:NEXT G
2200 Q=Q-1:FOR G=10 TO N:Q=Q+5
2210 IF Q>32 THEN PRINT :PRINT :Q=Q-35
2220 PRINT TAB(Q);G;:NEXT G
2230 CURSOR5,20:INPUT "Another Month Y/N ?";D$
2240 IF D$="Y"GOTO 2000
2250 CLS:PRINT :INPUT "Return to menu Y/N ?";S$
2260 IF S$="Y"GOTO 200
2270 IF S$<>"N" THEN PRINT "Invalid Response!":GOTO 2230
2280 IF S$="N"GOTO 400
3000 REM  DAY OF WEEK ROUTINE
3010 CLS:FOR X=1 TO 50:NEXT X:PRINT "   Enter Date (in numbers)":PRINT :PRINT
3020 INPUT "    The YEAR....";Y
3030 IF Y<1753 THEN PRINT :PRINT "Year must be after 1752'":GOTO 3020
3040 INPUT "    The MONTH...";M
3050 IF M<1ORM>12 THEN PRINT :PRINT "Invalid Month!":GOTO 3040
3060 INPUT "    The DAY.....";D
3070 LY=0:IF (YMOD4=0 AND YMOD100<>0) OR(YMOD400=0) THEN LY=1
3080 N=N(M):IF M=2AND LY=1 THEN N=N+1
3090 IF D<1ORD>N THEN PRINT "Invalid Date!":PRINT :GOTO 3060
3100 GOSUB 1000
3110 CLS:CURSOR9,5:PRINT D;",";M;",";Y
3120 PRINT :PRINT TAB(10);"IS A..";J$(Z)
3130 PRINT :PRINT :PRINT
3140 CURSOR5,20:INPUT "Another date Y/N ?";D$
3150 IF D$="Y"GOTO 3010
3160 CLS:PRINT :INPUT "Return to menu Y/N ?";S$
3170 IF S$="Y"GOTO 200
3180 IF S$<>"N" THEN PRINT "Invalid response!":GOTO 3140
3190 IF S$="N"GOTO 400
4000 REM HOLIDAY ROUTINES
4010 CLS:FOR X=1 TO 50:NEXT X:PRINT :PRINT
4020 INPUT "     Year required ?";Y
4030 IF Y<1960 THEN PRINT :PRINT "The Holidays shown are based on       present
day and may not be entirely correct.":FOR X=1 TO 1000:NEXT X
4040 CLS:PRINT "     The Holidays for";Y
4050 PRINT       "     "
4060 REM NEW YEAR
4070 D=1:M=1:GOSUB 1000
4080 IF Z=7 THEN Z=2:D=3
4090 IF Z=1 THEN Z=2:D=2
4100 PRINT "New Year...";TAB(26-LEN(J$(Z)));J$(Z);",";D;",";M$(1)
4110 IF Z=6 THEN Z=1:D=3
4120 PRINT TAB(22-LEN(J$(Z+1)));"and ";J$(Z+1);",";D+1;",";M$(1)
4130 REM WELLINGTON ANNIVERSARY
4140 D=22:GOSUB 1000
4150 Z=Z-2:IF Z>3 THEN D=D+7-Z
4160 IF Z<4 THEN D=D-Z
4170 PRINT :PRINT "Anniversary Day... ";J$(2);",";D;",";M$(1)
4180 REM  WAITANGI DAY
4190 D=6:GOSUB 1000
4200 PRINT :PRINT "Waitangi Day...";TAB(25-LEN(J$(Z)));J$(Z);","; 6;",";M$(2)
4210 REM  EASTER
4220 GOSUB 1100
```

```
4230 W=M:V=D-2:IF V<0 THEN V=31-V:W=M-1
4240 PRINT :PRINT "Good Friday...        ";J$(6);",";V;",";M$(W)
4250 W=M:V=D+1:IF V>31 THEN V=V-31:W=M+1
4260 PRINT "Easter Monday...      ";J$(2);",";V;",";M$(W)
4270 REM ANZAC DAY
4280 D=25:M=4:GOSUB 1000
4290 PRINT :PRINT "Anzac Day...";TAB(27-LEN(J$(Z)));J$(Z);", 25,";M$(4)
4300 REM SOVEREIGN'S BIRTHDAY
4310 D=1:M=6:GOSUB 1000
4320 IF Z=1 THEN D=2
4330 IF Z>2 THEN D=10-Z
4340 PRINT :PRINT "Sovereign's Birthday...";J$(2);",";D;",";M$(6)
4350 REM LABOUR DAY
4360 D=22:M=10:GOSUB 1000
4370 IF Z=1 THEN D=23
4380 IF Z>2 THEN D=31-Z
4390 PRINT :PRINT "Labour Day...       ";J$(2);",";D;",";M$(10)
4400 REM CHRISTMAS
4410 D=25:M=12:GOSUB 1000
4420 IF Z=7 THEN Z=2:D=27
4430 IF Z=1 THEN Z=2:D=26
4440 PRINT :PRINT "Christmas day..";TAB(24-LEN(J$(Z)));J$(Z);",";D;",";M$(12)
4450 D=26:IF Z=6 THEN Z=1:D=27
4460 PRINT "Boxing Day..";TAB(24-LEN(J$(Z+1)));J$(Z+1);",";D;",";M$(12)
4470 PRINT :INPUT " Another Year Y/N ?";D$
4480 IF D$="Y"GOTO 4000
4490 CLS:PRINT :INPUT "Return to Menu Y/N ?";S$
4500 IF S$="Y"GOTO 200
4510 IF S$<>"N" THEN PRINT "Invalid Response!":GOTO 4470
4520 IF S$="N"GOTO 400
```

ERROR MESSAGES

There is one noise that sends a shiver down the spine of any SEGA owner. (Not Philip Kenyon's voice which has been known to bring on attacks of nausea). It's that awful BEEP 2. Here, here (MH)! All programmers and users live in perpetual fear of having that spring upon them while running a program, so here is a list of how to avoid, detect and correct errors.

What to do when you are given an error message:

1. Don't panic!
2. Don't give up.
3. List the line of the program mentioned in the error message.

Syntax Errors:

These are just plain spelling mistakes. You have keyed in the wrong information or something that the computer doesn't understand. Check the line thoroughly character by character watching to see that you have not used an i instead of a number one or a 0 instead of a number zero or vice-versa. If the computer lists two lines of the program when you have only asked for one the problem is that you have not pressed enter at the end of the first line.

Solution: To correct Syntax errors either fix any visible mistakes or re-type the entire line. If two lines are joined together, remove the second line and then press CR. You will have to re-type the second line.

Statement and Function Parameter Errors:

These are a little harder to fix. What these errors mean is that one of the variables (e.g., x, y, a, b$, z$) used some where on that line is too big or to small. The mistake is probably not on this line. It will have occured earlier on in the program. Check each variable in the line by PRINTing it on the screen. Then when you have found the variable which you think is too big or too small check back through the program looking for any lines that use this variable.

Out of Data Errors;

This occurs when READ and DATA are used. The computer has been told to read a set number of pieces of data and there are none left! The poor dumb computer can't accept this sort of human oversight so it gives up and produces an error. Check through all the data making sure it's all there. Then make sure that all the RESTORE commands are in the right place.

Type Mismatch Error:

You will get this error when you try to get the computer to accept letters instead of a number. In other words you are trying to put a string (letters) into a number variable (which can only be numbers).

This normally happens with READ and DATA when a letter is mistakenly entered into the data lines of the program. To fix this error if it occurs with READ and DATA check the Data. If it occurs else where make sure you have not forgotten any string ($) signs in that line and further back in the program.

Undefined Line Number Error:

This happens with GOTO and GOSUB. You have told the computer to jump to a line number that does not exist. List the program around the area the computer was told to goto and check that no lines are missing. Make sure the line number you have used with your GOTO or GOSUB is right.

How to prevent errors:

1. Be more careful!
2. When copying programs out of a book get ruler or a piece or paper and use it as a guide under the line you are typing in. This means it's a lot harder to lose your place and skip lines.
3. Redifine the number zero by entering:

```
PATTERN
C # 48,"7Ø8898A8C8887ØØØ"
```

This places a diagonal line across the middle of the zero making it easier to distinguish from the letter O.

MAIL ORDER & SAVE!

ON SEGA AND AMSTRAD

FREE OVERNIGHT DELIVERY N.Z. WIDE!

OR SHOP AT OUR STORE

manukau COMPUTERS (N.Z.) LTD.

TOTAL RANGE OF SOFTWARE!

We have what you need:

- Printer Paper & Pens
- Disks
- Cords, Leads, Etc.
- Everything In Stock!

WRITE FOR A COMPLETE PRICE LIST!

NO CLONES! Beware of cheap imitations! We sell only TOP BRANDS—At Competitive Prices!

AMSTRAD
ONE GREAT IDEA AFTER ANOTHER

- 64K OF RAM (42K available)
- 32K of ROM
- Ready-to-go—JUST PLUG IT IN!
- High resolution graphics
- STEREO SOUND 3-voice, 7-octave stereo output fed through a hi-fi amplifier and speakers
- Expanding range of Software
- Unlimited scope for expansion

BUILT—IN PARALLEL PRINTER INTERFACE
OPTIONAL DISK DRIVE SYSTEM INCLUDING CP/M AND LOGO
JOYSTICK PORT

SEGA

- 32K COMPUTER $495
- MONITOR $595
- DISK DRIVE $995
- JOYSTICK $29.95
- DATASETTE $89.95
- PRINTER $495

Credit Card Phone Orders (Auckland 09) 656-002

HOURS: 10-5 Mon.—Sat.

P.O. Box 26-074, Auckland 3
Corner Manukau and Pah Roads, Epsom

NAME _____

ADDRESS _____

Credit Card No. _____ *Exp. Date* _____

- ☐ ARMSTRAD GREEN SCREEN @ $995
- ☐ ARMSTRAD COLOUR SCREEN @ $1395
- ☐ ARMSTRAD DISK DRIVE @ $795
- ☐ ARMSTRAD (SECOND) DISK DRIVE @ $595
- ☐ SEGA 32K COMPUTER $495
- ☐ SEGA MONITOR $595
- ☐ SEGA DISK DRIVE $995
- ☐ JOYSTICK $29.95
- ☐ SEGA PRINTER $495
- ☐ SEGA DATASETTE $89.95

TO THE DOOR RETURN ON ANY WARRANTY SERVICE—DIRECT TO US

☐ SEND ME A **FREE** PRICE LIST

AUSTRALIAN CUSTOMERS NOTE: FAVOURABLE EXCHANGE RATE AND NO SALES TAX MAKE THESE PRICES EVEN MORE INCREDIBLE VALUES! **PLEASE WRITE FOR MORE INFORMATION**

GLOSSARY

Accessory Devices - additional equipment which attaches to the computer and extends its functions and capabilities. Included are preprogrammed cartridges* and units which send, receive or store computer data, such as printers and disks. These are often called peripherals.

Array - A collection of numeric or string variables, arranged in a list or matrix for processing by the computer. Each element in an array is referenced by a subscript* describing its position in the list.

ASCII - The American Standard Code for Information Interchange, the code structure used internally in most personal computers to represent letters, numbers, and special characters.

BASIC - an easy-to-use popular programming language used in most personal computers. The word BASIC is an acronym for "Beginners All purpose Symbolic Instruction Code."

Baud - commonly used to refer to bits per second.

Binary - a number system based on two digits, 0 and 1. The internal language and operations of the computer are based on the binary system.

Branch - a departure from the sequential performance of program statements. An unconditional branch causes the computer to jump to a specified program line every time the branching statement is encountered. A conditional branch transfers program control based on the result of some arithmetic or logical operation.

Breakpoint - a point in the program specified by the STOP command where program execution can be suspended. During a breakpoint, you can perform operations to help you to locate program errors. Program execution can be resumed with a CONT command, unless editing took place while the program was stopped.

Bug - a hardware defect or programming error which causes the intended operation to be performed incorrectly.

Byte - a string binary* digits (bits) treated as a unit, often representing one data character*. The computer's memory capacity is often expressed as the number of bytes available. For example, a computer with 16K bytes of memory has about 16,000 bytes available for storing programs and data.

Cartridges - preprogrammed ROM* modules which are easily inserted in the SEGA computer to extend its capabilities.

Character - a letter, number, punctuation symbol, or special graphics symbol.

Command - an instruction which the computer performs immediately. Commands are entered with no preceding line number.

Concatenation - linking two or more strings* to make a longer string. The "+" is the concatenation operator.

Constant - a specific numeric or string* value. A numeric constant is any real number such as 1.2 or -9054. A string constant is any combination of up to 248 characters enclosed in quotes, such as "HELLO THERE" or "275 FIRST ST."

Cursor - a symbol which indicates where the next character* will appear on the screen when you press a key.

Data - basic elements of information which are processed or produced by the computer.

Default - a standard character or value which the computer assumes if certain specifications are omitted within a statement* or a program*.

Device - (see Accessory Devices)

Disk - a mass storage device capable of random and sequential access.

Display - (noun) the video screen; (verb) to cause characters to appear on the screen.

Execute - to run a program; to perform the task specified by a statement* or command*.

Exponent - a number indicating the power to which a number or expression* is to be raised; usually written at the right and above the number. For example, $2^8 = 2x2x2x2x2x2x2x2$. In SEGA BASIC the exponent is entered following the letter "E" in scientific notation*. For example, $2^8 = 1.3 \times 10^8 = 1.3E25$.

Expression - a combination of constants, variables, and operators which can be evaluated to a single result. Included are numeric, string, and relational expressions.

File - a collection of related data records stored on a device; also used interchangeably with device* for input/output equipment which cannot use multiple files, such as a line printer.

Function - a feature which allows you to specify as "single" operations a variety of procedures, each of which actually contains a number of steps; for example, a procedure to produce the square root via a simple reference name.

Graphics - visual constructions on the screen, such as graphs, patterns, and drawings, both stationary and animated. SEGA BASIC has build-in subprograms which provide easy to-use colour graphic capabilities.

Hardware - the various devices which comprise a computer system, including memory, the keyboard, the screen, disk drives, line printers, etc.

Hertz(HZ) - a unit of frequency. One Hertz = one cycle per second.

Hexadecimal - a base 16 number system using 16 symbols, 0-9 and A-F. It is used as a convenient "shorthand" way to express binary* code. For example, 1010 in binary = A in hexadecimal, 11111111 = FF. Hexadecimal is used in constructing patterns for graphics characters in the PATTERN subprogram.

Increment - a positive or negative value which consistently modifies a variable*.

Input - (noun) data* to be placed in computer memory; (verb) the process of transferring data into memory.

Input Line - the amount of data* which can be entered at one time. In SEGA BASIC, this is 248 characters.

Internal date-format - data* in the form used directly by the computer. Internal numeric data is 8 bytes* long plus 1 byte which specifies the length. The length for internal string data is one byte per character in the string* plus one length-byte.

Integer - a whole number, either positive, negative or zero.

I/O - Input/Output; usually refers to a device function. I/O is used for communication between the computer and other devices (e.g. keyboard, disk)

Iteration - the technique of repeating a group of program statements; one repetition of such a group. See Loop.

Line - see input line, print line, or program line.

Loop - a group of consecutive program lines which are repeatedly performed, usually a specified number of times.

Mantissa - the base number portion of a number expressed in scientific notation*. In 3.264E+4, the mantissa is 3.264.

Mass Storage Device - an accessory device*, such as a cassette recorder or disk drive, which stores programs and/or data* for later use by the computer. This information is usually recorded in a format readable by the computer, not people.

Memory - see RAM, and ROM, and mass storage device.

Noise - various sounds which can be used to produce interesting sound effects. A noise, rather than a tone, is generated by the SOUND subprogram* when commands 4 and 5 are specified.

Null String - a string* which contains no characters and has zero length.

Number Mode - the mode assumed by the computer when it is automatically generating program line* numbers for entering or changing statements.

Operator - a symbol used in calculations (numeric operators) or in relationship comparisons (relational operators). The numeric operators are +,-,*,/,. The relational operators are <,<=,=>,>,=

Overflow - the condition which occurs when a rounded value greater than 9.9999999999999E+99 or less than -9.9999999999999E-99 is entered or computed. When this happens, a warning is displayed, and the program* stops.

Output - (noun) information supplied by the computer; (verb) the process of transferring information from the computer's memory onto a device, such as a screen, line printer, or mass storage device*.

Parameter - any of a set of values that determine or affect the output of a statement* or function*.

Print Line - a 38-position line used by the PRINT statement.

Program - a set of statements which tell the computer how to perform a complete task.

Program Line - a line containing a single statement*. The maximum length of a program line is 248 characters*.

Pseudo-random number - a number produced by a definite set of calculations (algorithm) but which is sufficiently random to be considered as such for some particular purpose. A true random number is obtained entirely by chance.

Software - various programs which are executed by the computer, including programs built into the computer. Cartridges* programs, and programs entered by the user.

Statement - an instruction preceded by a line number in a program. In SEGA BASIC, more than one statement is allowed in a program line*.

RAM - random access memory; the main memory where program statements and data* are temporarily stored during program execution*. New programs and data can be read in, accessed, and changed in RAM. Data stored in RAM is erased whenever the power is turned off of BASIC is exited.

Reserved Word - in programming languages, a special work with a predefined meaning. A reserved word lmust be spelled correctly, appear in the proper order in a statement* or command*, and cannot be used as a variable* name.

ROM - read-only memory; certain instructions for the computer are permanently stored in ROM and can be accessed but cannot be changed. Turning the power off does not erase ROM.

Run Mode - when the computer is executing* a program, it is in Run Mode. Run Mode is terminated when program execution ends normally or abnormally. You can cause the computer to leave Run Mode by pressing BREAK during program execution (see Breakpoint*).

Scientific Notation - a method of expressing very large or very small numbers by using a base number (mantissa*) times ten raised to some power (exponent*). To represent scientific notation in SEGA BASIC enter the sign, then the mantissa, the letter E, and the power of ten (preceded by a minus sign if negative). For example, 3,264; -2.47E -17.

Scroll - to move the text on the screen so that additional information can be displayed.

String - a series of letters, numbers and symbols treated as a unit.

Subprogram - a predefined general-purpose procedure accessible to the user through the statement in SEGA BASIC. Subprograms extend the capability of BASIC and cannot be easily programmed in BASIC.

Subroutine - a program segment which can be used more than once during the execution* of a program, such as a complex set of calculations of a print routine. In SEGA BASIC a subroutine is entered by a GOSUB statement and ends with a RETURN statement.

Subscript - a numeric expression which specifies a particular item in an array*. In SEGA BASIC the subscript is written in parentheses immediately following the array name.

Underflow - the condition which occurs when the computer generates a numeric value greater than -1E-100, less than 1E-100, and not zero. When an underflow occurs, the value is replaced by zero.

Variable - a name is given to a value which may vary during program execution. You can think of a variable as a memory location where values can be replaced by new values during program execution.

***See definition in Glossary**

CHANGE OF ADDRESS

Name .....

Old Address	New Address
.....	.....
.....	.....

Date of change of address .....

GRANDSTAND LEISURE LIMITED

If you are having any problems obtaining any of the pieces of software please contact Granatand Leisure enclosing a cheque or postal order.

SOFTWARE

CARTRIDGES	PRICE
☐ BASIC LEVEL IIIB	$175
☐ MUSIC	$125
☐ N/SUB	$39.95
☐ SAFARI HUNTING	$39.95
☐ BORDER LINE	$39.95
☐ CONGO BONGO	$39.95
☐ YAMATO	$39.95
☐ STAR JACKER	$39.95
☐ CHAMPION TENNIS	$39.95
☐ MONACO GP	$39.95
☐ CHAMPION BASEBALL	$39.95
☐ SINBAD MYSTERY	$39.95
☐ VIDEO FLIPPER	$39.95
☐ PACAR	$39.95
☐ POP FLAMER	$39.95
☐ CHAMPION GOLF	$49.95
☐ EXERION	$49.95
☐ SAFARI RACE	$49.95
☐ LODE RUNNER	$49.95
☐ BOXING	$54.95
☐ ORGUS	$49.95
☐ FLICKY	$59.95

HOME/PERSONAL/FINANCE CASSETTES	PRICE
☐ FILE SYSTEM	$19.95
☐ CHEQUE BOOK RECONCILIATION	$19.95
☐ LOAN & MORTGAGE CALCULATION	$19.95
☐ ACCOUNTS RECEIVABLE	$30.00
☐ ACCOUNTS PAYABLE	$30.00
☐ 32K WORD PROCESSOR	$30.00
☐ PERSONAL RECORD KEEPER	$30.00
☐ 16K EASY WRITER	$19.95

GAME CASSETTES	PRICE
☐ TOWERS OF HANOI	$19.95
☐ HANGMAN	$19.95
☐ CITY-LANDER	$19.95
☐ CUBE-IT	$30.00
☐ BUGALOO	$19.95
☐ LASER BLAST	$19.95
☐ MUNCHMAN	$30.00
☐ ENTERPRISE ESCAPE	$19.95
☐ MARSMOBILE	$19.95
☐ MARS ADVENTURER	$19.95
☐ EMPIRE	$19.95
☐ KALAH	$19.95
☐ CHESS	$19.95
☐ DEATH SATELLITE	$19.95
☐ BIT BYTE	$19.95
☐ THE HOUSE	$30.00
☐ THE PYRAMID	$19.95
☐ CODE BREAKER	$19.95
☐ BACKGAMMON	$19.95
☐ LUMBER JACK	$19.95
☐ TRAMPMAN	$19.95
☐ SEGA GAMES PACK I	$19.95

EDUCATIONAL CASSETTES	PRICE
☐ ROCKET MATHS	$19.95
☐ LEARNING ALPHABET	$19.95
☐ WATCH ME DRAW	$19.95
☐ SPELLING	$19.95
☐ ADDITION	$19.95
☐ SUBTRACTION	$19.95
☐ MULTIPLICATION	$19.95
☐ SPRITE EDITOR	$19.95
☐ MUSIC EDITOR	$19.95
☐ TYPING TUTOR	$19.95
☐ LEARNING TO COUNT	$19.95
☐ SHAPE & COLOUR QUIZ	$19.95
☐ MUSIC CARTRIDGE DEMONSTRATOR	$19.95
☐ SEGA GRAPHIC DESIGNER	$19.95
☐ GEOGRAPHY QUIZ I	$19.95
☐ GEOGRAPHY QUIZ II	$19.95
☐ MATHS HANG UP	$19.95

I enclose payment

☐ Cash ☐ Cheque ☐ Postal order ☐ Visa ☐ Bankcard

NAME..... CARD NUMBER

ADDRESS..... | | | | | | | | | | | | | | | | |

..... CARD EXPIRY DATE.....

SIGNATURE... CHEQUE NO.....

GRANDSTAND
P.O. BOX 2353
AUCKLAND

Two Programs at Once

by Michael Howard

The following machine-code allows you to have two individual programs in RAM at once!

To run the program firstly enter the machine code data. Run. Then type NEW, don't worry the program is safe. Now write write a small program say:

```
10 REM TEST #1
20 A$="THIS IS A TEST"
30 FOR A=1 TO LEN(A$):PRINT LEFT$(A$,A)
40 NEXT A
```

Run it, now type CALL$FØØD (This executes the machine code), now LIST. You'll see that the program has gone! Next enter another program say:

```
10 REM TEST #1
20 B$="THIS IS SILLY!!"
30 FOR B=1 TO LEN(B$)
40 PRINT RIGHT$(B$,B)
50 NEXT B
```

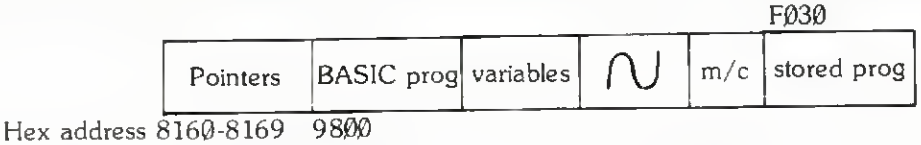
Run the program, now type CALL$FØØD (This once more activates the machine code), now list, the 1st program is back! Run it! It works okay. Even the variables (A$&B$,A&B) have swapped over! The only draw-back is that only 198Ø bytes of program can be swapped, so when memory left gets to 1Ø34 bytes. . .STOP, if you continue your program will get chopped off!

Technical Info.

Disassembly org &HFØØØ (org- means start of machine code)

```
1,98,2,98,3,98          ,data for pointers, end of prog, start/end of variables
lda(hl)
exaf,a f'
ld a (de)
ld(hl),a                swap (hl)      (de)
exaf,a'f'               uses alternate a register (a')
ld(de),a
ret
ld hl&8162
ld de $HFØØØ
ld b, 6
call &HFØØ6 ←           exchange pointers
inchl
incde
djnz ——
ldbc&HØ7CØ
ld hl &H98ØØ
call &HFØØ6 ←           swap program
inc hl                  bc — counter
inc de                  hl — start of BASIC program
dec bc                  de — start of BASIC program (stored program)
lda, b
orc
jrnz ——
ret
```

Normally a BASIC program is stored from &H98ØØ onwards, the top of program is governed by a 16-bit address held in &H8162 and&H8163.The stored program is held in address &HFØ3Ø onwards. All BASIC programs end in two carriage returns (&HD and &HD). When the programs are swapped over the pointers (end of program, beginning of variables and end of variables) are poked into place first followed by the actual program. If this all seems mumbo-jumbo just look at the memory map:



When the machine code is used (at &HFØØØ) the stored program and BASIC program swap places.

Dear Sega User Club Member

As a valued member of the Sega Users Club, we would like to offer you a chance to buy the bargain of the year. An ideal birthday gift, or buying early for Christmas.

The Sega SG1000 plus your choice of game worth $39.95 (see below) for only $89.95 including power supply and one joystick.

This fantastic unit plays all the superb Sega game cartridges when connected to your own T.V, in exactly the same way as the computer. Later on you could upgrade the unit to a full computer with the add on keyboard, or trade in against the SC3000, and keep all your game cartridges.

This offer must be limited to just 2 units per member, and all payments must be received within 3 weeks to guarantee supply.

To take advantage of this offer send your cheque payable to Grandstand Leisure Ltd with the return slip below.

I would like to buy. . . . . special offer Sega SC1000 games computer with the indicated free game/s.

N SUB	☐
YAMATO	☐
CHAMPION	☐
BASEBALL	☐
VIDEO FLIPPER	☐
POP FLAMER	☐

Post to Grandstand Leisure Ltd
C.P.O. Box 2353
AUCKLAND 1


STILL AVAILABLE
TEACH YOURSELF
BASIC PROGRAMMING
FOR USE WITH 16 OR 32K SEGA
SEGA®
Beginners Guide
Practical advice on all aspects of using the Sega Computer Peripheral attachments, and software takes the frustration out of getting to know your SEGA COMPUTER.
GRANDSTAND
SEGA

The Hord Word from Sego

SEGA'S new hard keyboard end new herd disk drive, make Sega even herder to beat.

The very letest industry stendard 3'' diskettes store e messive 328K of information eech desk, end operate at speeds fer in excess of their older 5½ inch counterperts.

Sege's super control station will also eneble you to connect to many other attachments such as business printers electronic typewriters even telephone moderns, through it's two additional 1/0 ports, for RS232C end parrallel interfaces.

Sega now boasts not only the most sophisticated and easy to learn basic language, the most incredible graphic capabilities, a sensational range of cartridge and cassette games, superb education and applications software written by and for New Zealanders, but also a very sophisticated, fast, small business system, with the state of the art Sega control station.
Feel the difference, see the difference, compare the difference, and you too can enjoy the difference of Sega superiority.
Call at any Sega stockist for a demonstration or contact Grandstand Leisure at: Box 2353 AUCKLAND

GRANDSTAND
SEGA®